VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Metody pro konstrukci komplexních sítí z multidimenzionálních dat

Methods for Construction of Complex Networks from Multidimensional Data

2020 Pavol Urban

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Pavol Urban**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Metody pro konstrukci komplexních sítí z multidimenzionálních dat
Methods for Construction of Complex Networks from Multidimensional Data

Jazyk vypracování: slovenština

Zásady pro vypracování:

Cílem práce je prostudovat metody pro konstrukci sítí z vektorových dat a seznámit se standardními i méně standardními přístupy, které se v této oblasti používají. Vybrané metody implementovat, dosažené výsledky interpretovat a vzájemně porovnat. Součásti práce je i možnost vizualizovat pouze redukovanou datovou sadu tak, aby byly zachovány charakeristické vlastnosti celé datové sady.

1. Prostudujte problematiku konstrukce komplexních sítí z multidimenzionálních dat.
2. Vyberte datové kolekce, předzpracujte je a vhodně uložte pomocí efektivních datových struktur.
3. Implementuje vybrané metody pro konstrukci sítí spolu s vhodnými metrikami pro výpočet vzdálenosti a podobnostmi.
4. Implementujte metody, které vytvoří redukovanou síť.
5. Experimentujte s implementovanými metodami, proveďte standardní analýzy vytvořených sítí.
6. Vhodně zvolte vizualizaci zkonstruovaných sítí a jejich analýz.

Seznam doporučené odborné literatury:

[1] Talukdar, Partha Pratim. "Topics in graph construction for semi-supervised learning." (2009).
[2] Ochodkova, Eliska, Sarka Zehnalova, and Milos Kudelka. "Graph construction based on local representativeness." International Computing and Combinatorics Conference. Springer, Cham, 2017.
[3] Fritzke, Bernd. "A growing neural gas network learns topologies." Advances in neural information processing systems. 1995.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Mgr. Pavla Dráždilová, Ph.D.**

Datum zadání: 01.09.2019

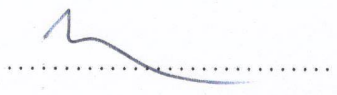
Datum odevzdání: 30.04.2020

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

V Ostrave 14. mája 2020



Týmto by som chcel poďakovať vedúcej diplomovej práce Mgr. Pavle Dráždilovej, Ph.D. za odbornú pomoc a konzultácie pri vytváraní tejto práce.

Abstrakt

Táto diplomová práca je zameraná na konštrukciu komplexných sietí z vektorových dát. Sú v nej popísané základné pojmy a vlastnosti z teórie grafov a tiež postupnosť krokov, ktorá umožňuje skonštruovať komplexné siete z vektorových dát. Táto postupnosť je implementovaná vo výslednej aplikácii. Vybrané dátové sady sú potom spracované touto aplikáciou a následne analyzované. V práci je tiež popísané vzorkovanie grafu, ktoré umožňuje vytvárať redukované siete.

Kľúčové slová: komplexné siete, multidimenzionálne dáta, vlastnosti sietí, metrika sietí, teória grafov, vzorkovanie grafu

Abstract

This thesis is focused on the construction of complex networks from multidimensional data. In this thesis are described the basic concepts and properties from the graph theory, as well as the sequence of steps that allows constructing complex networks from multidimensional data. This sequence is implemented in the application. The selected data sets are then processed by this application and subsequently analyzed. This thesis also describes graph sampling, which allows creating reduced networks.

Key Words: complex networks, multidimensional data, network properties, network metrics, graph theory, graph sampling

Obsah

Zoznam použitých skratiek a symbolov

FR Layout	–	Fruchterman-Reingold Layout
GDPR	–	General Data Protection Regulation
GUI	–	Graphical User Interface
HBSC	–	The Health Behavior in School-aged Children
JUNG	–	Java Universal Network/Graph Framework
KK Layout	–	Kamada-Kawai Layout
k-NN	–	K-Nearest Neighbors Algorithm
LRNet	–	Local Representativeness Network
RDN	–	Random Degree Node
RE	–	Random Edge
RN	–	Random Node
UX	–	User Experience
WHO	–	World Health Organization

Zoznam obrázkov

Zoznam tabuliek

Zoznam výpisov zdrojového kódu

1 Úvod

Či už si to uvedomujeme alebo nie, siete využívame denne. Ak cestujeme, využívame cestnú, železničnú či leteckú sieť [28]. Ak komunikujeme s ľuďmi, využívame siete našich kontaktov [22, 30]. Fenoménom sú, samozrejme, sociálne siete [5, 6, 7, 17, 32]. Všetky z týchto zmienených sietí reprezentujú najrozličnejšie vzťahy medzi objektmi v mnohých aspektoch života. Medzi najväčšie výhody sietí patrí ich vizualizácia a možnosť analyzovať ich. Vizualizáciou získavame prvotný obraz o vlastnostiach siete. Analýzou sietí potom získavame detailnejšie vedomosti o vlastnostiach siete, ktoré nám umožňujú lepšie pochopenie vzťahov v týchto sieťach. Na takúto analýzu bolo vyvinutých množstvo nástrojov a knižníc, ako sú napríklad [1, 2, 4, 21, 23] a mnoho ďalších.

Tieto nástroje a knižnice môžu nájsť uplatnenie aj pri analýzach dát, ktoré ako také sieť netvoria. Veľmi zaujímavým príkladom takýchto dát sú vektorové dáta. Vektorové dáta sú tvorené jednotlivými záznamami (položkami, riadkami či objektmi), ktoré môžu mať ľubovoľný počet atribútov (stĺpcov). Príkladom vektorových dát môže byť nákupný zoznam či štatistika futbalových zápasov.

Keďže jednou z nevýhod vektorových dát je ich problematická vizualizácia (pri vyššom počte dimenzií), ktorá je naopak jednou z najväčších výhod sietí, práve siete môžu byť vhodným spôsobom ako reprezentovať vektorové dáta. V tejto práci je popísaná celá postupnosť krokov, ktorá umožňuje vytvoriť siete z vektorových dát.

V úvodnej kapitole sa pozrieme na možnosti prevedenia vektorových dát na siete, kde si tiež vysvetlíme prácu s nekompletnými záznamami, výpočty vzdialeností a podobností medzi jednotlivými záznamami vektorových dát, či možnosti vizualizácie siete. V tejto kapitole si taktiež popíšeme možnosti vytvárania redukovaných sietí z pôvodných sietí a vysvetlíme si, v čom spočívajú ich výhody a nevýhody. V ďalšej kapitole si popíšeme základné vlastnosti sietí vychádzajúce z teórie grafov. V štvrtej kapitole si stručne popíšeme technológie, ktoré boli použité pri implementácii vlastného nástroja. V piatej kapitole si bližšie popíšeme dátové sady, ktoré boli využité na testovanie a experimentovanie. Primárne ide o dáta vychádzajúce z HBSC štúdie, ktorá sa zaoberá správaním školopovinných detí v súvislostim so zdravím a v spomínanej kapitole sa bližšie dočítame aj o štúdii samotnej. V nasledujúcej kapitole nájdeme popísaný vlastný nástroj a možnosti jeho ovládania. V poslednej kapitole si potom popíšeme výsledky experimentovania s vybranými atribútmi spomínaných dátových sád.

2 Od vektorových dát k sieti

Táto práca sa zameriava na vytváranie a následnú analýzu komplexných sietí, kde vstupné dátové sady sú vo forme vektorových dát. Proces prechodu od vektorových dát k sieti je realizovaný postupnosťou týchto krokov:

1. spracovanie chýbajúcich údajov,
2. normalizácia dát,
3. metódy výpočtu vzdialeností,
4. metódy prevodu vektorových dát na siete,
5. metódy vzorkovania sietí,
6. vizualizácia siete.

Tieto kroky nie sú viazané na konkrétny dataset, ale sú akousi univerzálnou postupnosťou, ktorá je použiteľná pre ľubovoľné vektorové dáta. Hoci práca je primárne zameraná na datasety z kapitoly 5, výsledná aplikácia umožňuje prevod ľubovoľných vektorových dát na sieť a jej následnú analýzu. Taktiež je potrebné uviesť, že aplikácia umožňuje vybrať ľubovoľnú podmnožinu atribútov vektorových dát, z ktorej bude tvorená výsledná sieť. To pre náš dataset z HBSC štúdie znamená to, že nemusíme používať všetky zo stoviek atribútov, ale umožňujeme sa zamerať len na vybrané konkrétne otázky (napríklad otázky z oblasti stravovania či pohybových aktivít). O datasetoch podrobnejšie pojednáva spomínaná kapitola 5.

2.1 Metódy na doplnenie chýbajúcich údajov

Problémom akýchkoľvek dát, ktoré by sme kedy mohli analyzovať, je to, že nie sú vždy kompletné. Pri výpočte vzdialeností a podobností takýto chýbajúci stĺpec znemožňuje výpočet, preto musíme zaviesť metódy ako s takýmito prázdnymi stĺpcami naložiť. V tejto práci budú použité tieto metódy na doplnenie chýbajúcich údajov, ktoré vychádzajú z [19] a [20]:

1. **vymazanie záznamu** - celý záznam, ktorý nemá hodnotu v príslušnom stĺpci z vybranej podmnožiny atribútov, bude odstránený, a teda vo výslednej sieti nebude mať uzol, ktorý by ho reprezentoval,
2. **doplnenie mediánu** - do chýbajúceho atribútu záznamu doplníme hodnotu mediánu príslušného stĺpca (medián je počítaný zo všetkých neprázdnych hodnôt v príslušnom stĺpci),
3. **doplnenie priemeru** - do chýbajúceho atritbútu záznamu doplníme hodnotu priemeru daného stĺpca (priemer je rovnako ako medián počítaný zo všetkých neprázdnych hodnôt v príslušnom stĺpci).

2.2 Normalizácia dát

Predtým ako si povieme o tom, ako môžeme spočítať vzdialenosť dvoch objektov, môžeme urobiť normalizáciu dát, ktorá nám pomáha vo chvíľach, ak sú jednotlivé atribúty z rôznych intervalov. V tejto práci bola použitá tzv. min-max normalizácia, ktorá zabezpečí, že hodnoty jednotlivých atribútov sa budú po normalizácii nachádzať v intervale <0,1>.

Túto normalizáciu vypočítame podľa vzťahu definovaného v [26] nasledovne:

$$z_i = \frac{x_i - min(X)}{max(X) - min(X)} \tag{1}$$

Kde $X = (x_1, ..., x_n)$ je vektor pôvodných hodnôt a $Z = (z_1, ..., z_n)$ je vektor normalizovaných hodnôt.

2.3 Metódy výpočtu vzdialenosti a podobnosti

V tejto kapitole sú uvedené vybrané metódy pre výpočet vzdialeností a podobností medzi dvoma prvkami (objektmi) reprezentovanými číselným vektorem. Vychádzame pritom z [9] a [15].

Nech $X = (x_1, ..., x_n)$ a $Y = (y_1, ..., y_n)$ sú vektory. Tieto vektory v našej práci tvoria jednotlivé riadky vektorových dát a budeme ich nazývať objektmi. V tejto práci si vzdialenosť dvoch objektov budeme značiť ako $d(X, Y)$. Poznáme viacero metód na výpočet vzdialeností. Poďme si predstaviť tie z nich, ktoré sú použité v tejto práci. Ako prvú si zadefinujeme tzv. euklidovskú vzdialenosť d_e, ktorá je jedna z najznámejších a je daná vzorcom:

$$d_e(X, Y) = \|X - Y\| = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{2}$$

Druhou metódou na výpočet vzdialenosti je tzv. manhattanovská daná vzťahom:

$$d_m(X, Y) = \sum_{i=1}^{n} |x_i - y_i| \tag{3}$$

Ďalšou metódou je Čebyševova vzdialenosť, ktorá je definovaná ako maximálna hodnota z absolutného rozdielu súradníc (keďže hľadáme maximálny rozdiel je dáto metóda občas označovená aj ako maximová). Zapisujeme:

$$d_c(X, Y) = max(|x_i - y_i|) \tag{4}$$

Ďalším spôsobom ako zistiť vzťah medzi dvoma objektmi, je takzvaný Pearsonov korelačný koeficient. Korelácia označuje vzťah medzi dvoma veličinami, teda nám umožňuje sledovať či sa

pri zmenej jednej veličiny mení aj druhá veličina. Korelačný koeficient si v tejto práci označíme ako r_{XY}. Vypočítame ho nasledujúcim vzťahom:

$$r_{XY} = \frac{\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y)}{\sqrt{\sum_{i=1}^{n}(x_i - \mu_x)^2}\sqrt{\sum_{i=1}^{n}(y_i - \mu_y)^2}} \tag{5}$$

Korelačný koefcient vo všeobecnosti môže nadobúdať hodnoty z intervalu <-1,1>. Hodnota 0 znamená, že objekty spolu nekorelujú, teda zjednodušene povedané, nie sú vo vzťahu. Hodnota 1 znamená maximálnu možnú pozitívnu koreláciu a hodnota -1 naopak maximálnu možnú negatívnu koreláciu. Pri korelácií v tejto práci uvažujeme len pozitívnu koreláciu, teda ako veľmi podobné boli dva objekty, čo umožňujeme úpravou hodnoty r_{XY} nasledovne:

$$r_{XY} = \begin{cases} r_{XY}, & (r_{XY} > 0); \\ 0, & (r_{XY} \leq 0). \end{cases} \tag{6}$$

Z uvedenej úpravy v rovnici 6 je zrejmé, že kladné hodnoty zachovávame a záporné položíme rovné nule. Touto úpravou sme docielili, že na r_{XY} môžeme nahliadať ako na podobnosť. Keďže tá je definovaná podľa [11] nasledujúcimi vlastnosťami:

1. s(X, Y) = s(Y, X),
2. s(X, X) 0,
3. s(X, X) $\geq$ s(X, Y),
4. s(X, Y) + s(Y, Z) $\leq$ s(X, Z) + s(Y, Y),
5. s(X, X) = s(Y, Y) = s(x, y) len a len vtedy, ak X = Y.

Na dátach využívaných v tejto práci, ktoré sú popísané v kapitole 5, nebola zistená negatívna korelácia.

Ďalšou možnosťou ako vypočítať vzťah medzi dvoma objektmi, je využiť takzvanú Gaussovu funkciu (tiež využívané je označenie RBF - radial basis function). Gaussovu funkciu podľa [38] vypočítame nasledovne:

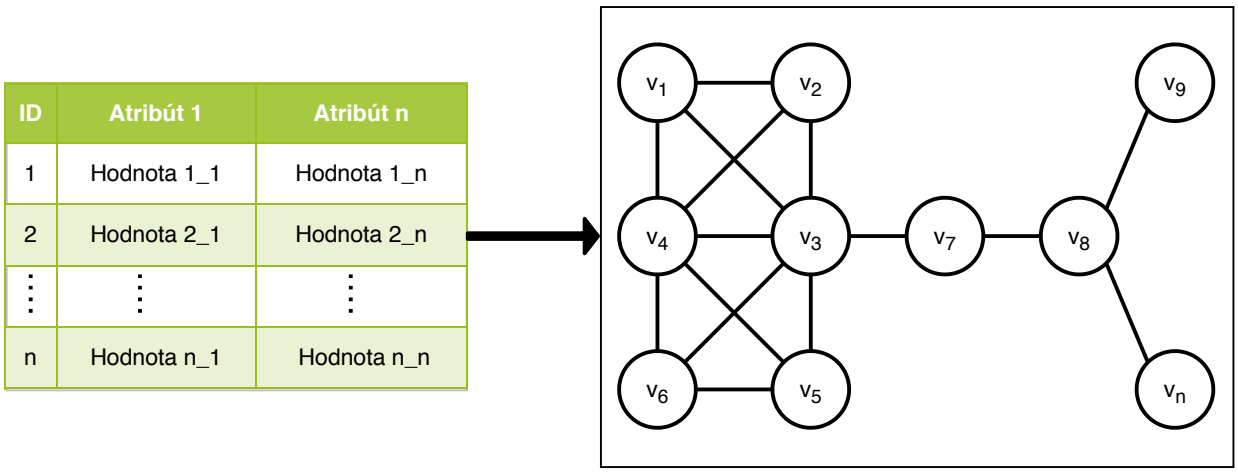
$$s_k(X, Y) = exp^{-\frac{d_e(X,Y)}{2\sigma^2}} \tag{7}$$

Kde σ je hyper-parameter a $d_e(X, Y)$ je euklidovská vzdialenosť, ktorú sme si už definovali v tejto kapitole. Keďže výsledná hodnota $s_k(X, Y) = 1$ pre vzdialenosť $d_e(X, Y) = 0$ a $s_k(X, Y)$ sa so zväčšujúcou sa vzdialenosťou $d_e(X, Y)$ limitne blíži k nule, Gaussova funkcia nám reprezentuje podobnosť medzi objektmi X a Y.

2.4 Metódy prevodu vektorových dát na siete

Vstupné dátové súbory sú v podobe vektorových dát. Táto práca má však za cieľ takéto dáta transformovať na sieť (pozri obrázok 1) a tak umožniť nahliadnuť na dáta práve z pohľadu sietí.

V nasledujúcich podkapitolách preto budú predstavené základné metódy, ktoré nám umožňuju vytvoriť siete z vektorových dát.

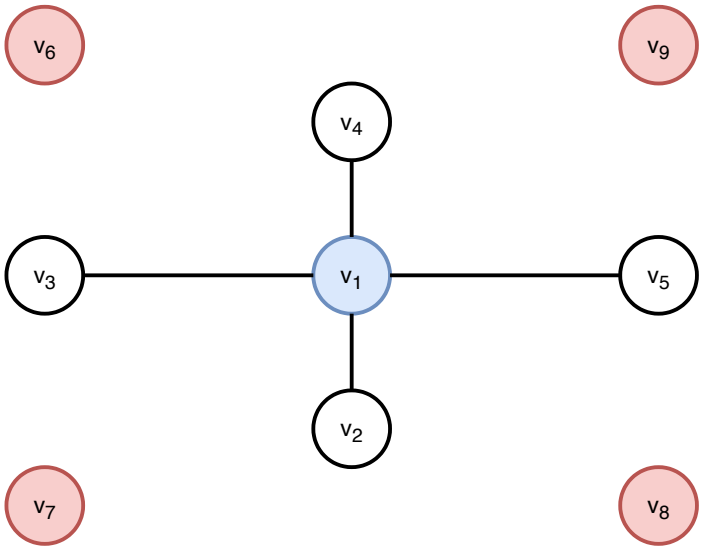


Obr. 1: Prevod vektorových dát na sieť

2.4.1 k-NN

Skratka k-NN vychádza z anglického výrazu k-nearest neighbors, čo môžeme preložiť ako k - najbližších susedov. Písmenom k označujeme premennú, ktorá určuje, koľko najbližších susedov budeme hľadať. Algoritmus funguje tak, že z každého jedného záznamu z vektorových dát vytvoríme uzol. Následne pre každý jeden uzol vypočítame z vektorových dát jeho vzdialenosť od všetkých ostatných uzlov a zaznamenáme k najbližších, ktoré pripojíme k danému uzlu hranou [38]. Teda v sieti bude každý uzol aspoň stupňa k.

Metódu k-NN si môžeme prezrieť na obrázku 2, kde k=4. Pre prehľadnosť bol ukázaný prevod len pre modro vyznačený uzol v_1, preto všetky uzly nemajú stupeň k. Na obrázku vidíme, že sa k uzlu v_1 pripojili 4 najbližšie uzly - v_2, v_3, v_4 a v_5, ktoré sú zobrazené bielou farbou. Červené uzly v_6, v_7, v_8 a v_9 nepatrili medzi 4 najbližšie k uzlu v_1, preto k nemu nie sú pripojené. Obmedzením tejto metódy je to, že vždy vyberie len k najbližších susedov a v podstate ignoruje ďalšie uzly, ktoré môžu byť taktiež veľmi blízko.



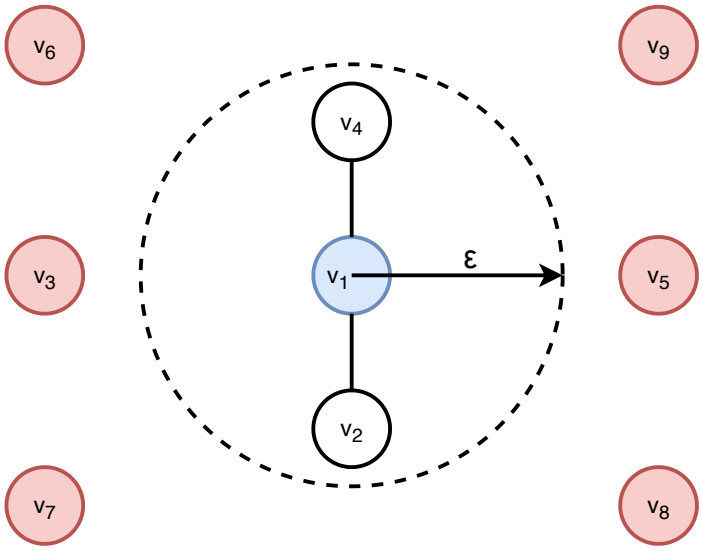
Obr. 2: Ukážka metódy k-NN (pre k=4)

2.4.2 ε - okolie

ε-okolie je metóda, pri ktorej podobne ako v k-NN, vytvoríme z každého jedného záznamu z vektorových dát uzol. Následne sa pre každý uzol počítajú vzdialenosti so všetkými ostatnými uzlami z vektorových dát. Ak je splnené, že vzdialenosť $d(XY) < \varepsilon$, tieto dva uzly budú prepojené [38]. V tomto prípade je premennou ε, jeho zväčšenie má za následok, že sieť bude hustejšia a ak ho zmenšíme, sieť bude obsahovať menej hrán (teda bude redšia).

Túto metódu si môžeme pozrieť na obrázku 3. ε-okolie sme rátali pre modrý uzol v_1. Biele uzly, ktoré sa nachádzali v ε-okolí vyznačeným prerušovanou čiarou sú susedmi uzla v_1. Červené uzly v_3, v_5, v_6, v_7, v_8 a v_9 boli za hranicou okolia a preto nie sú susedmi uzla v_1. Pre porovnanie pozri obrázok 2, kde uzly v_3 a v_5 boli susedmi uzla v_1.

Samozrejme s premennou ε môžeme manipulovať. Ak by sme nastavili ε rozumne, vedeli by sme v tomto konkrétnom prípade docieliť rovnakého výsledku ako na obrázku 2. Táto metóda má však problém najmä s outliermi (odľahlými pozorovaniami), pretože pri nich nepomôže ani zväčšovať ε a to z toho dôvodu, že by sme ho museli zväčšiť tak výrazne, až by veľká časť ostatných uzlov v sieti bola nesmierne husto poprepájaná. Túto nevýhodu sa snaží vyriešiť metóda, ktorá je rozobraná v nasledujúcej podkapitole.

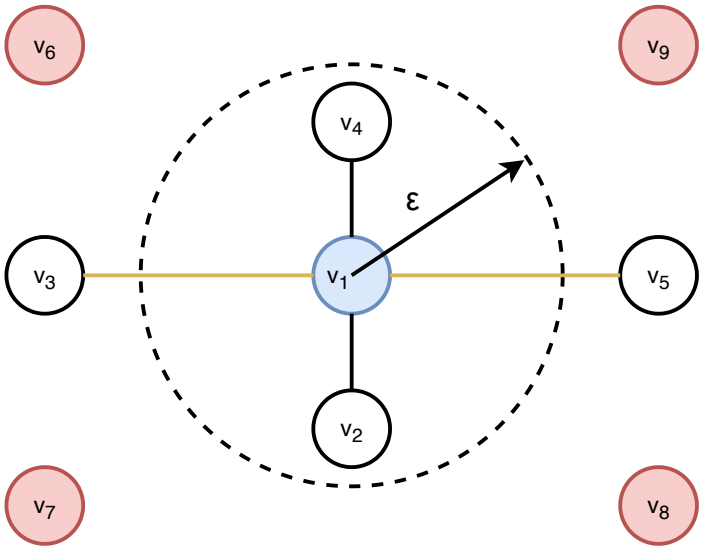


Obr. 3: Ukážka metódy ε-okolie

2.4.3 Kombinácia metód ε - okolie a k-NN

Metódy, ktoré sme si predstavili v predošlých dvoch podkapitolách, majú svoje nevýhody. Tie sa snaží vyriešiť algoritmus, ktorý je ich spojením. Začneme tým, že vytvoríme sieť podľa ε-okolia, tak ako sme si to predstavili v 2.4.2. Následne prejdeme všetky uzly ešte raz a kontrolujeme ich stupeň. Ak je nižší ako k, doplníme najbližšie uzly tak, aby mal daný uzol k susedov.

Zamerajme sa na obrázok 4 a porovnajme ho s obrázkom 3. Vidíme, že sme začali rovnako a za susedov sme označili uzly v_2 a v_4. Rozdiel je v tom, že sme nastavili premennú $k = 4$, a tak pribudli spojenia s uzlami v_3 a v_5 (vyznačené žltou čiarou). To v tomto konkrétnom prípade má za následok, že vznikli rovnaké prepojenia, ako pri použití samostatnej metódy k-NN (pozri obrázok 2. V praxi to znamená, že uzly, ktoré majú vo svojom okolí veľa blízkych uzlov, ich budú mať všetky ako susedné uzly (nie sú obmedzené len na vybrané k) a uzlom, ktoré vo svojom okolí nemajú aspoň k susedov, pribudne taký počet susedov, aby ich bolo aspoň k. Teda nie je možnosť, aby sieť obsahovala uzly stupňa 0, ako sa to môže stať pri samostatnej metóde ε-okolie. Toto všetko zvyšuje šancu, že sieť bude súvislá, hoci stále môžu vzniknúť pripády, keď tomu tak nebude.



Obr. 4: Ukážka kombinovanej metódy ε-okolie a k-NN (k = 4)

2.4.4 LRNet

Názov LRNet vychádza z anglického Local Representativeness Network, čo môžeme voľne preložiť ako sieť lokálnej reprezentatívnosti. Metóda LRNet, tak ako bola predstavená v [34], nahliada na najbližších susedov iným spôsobom ako v metódach predstavených v predchádzajúcich odsekoch. Predpokladáme, že objekty - v našom prípade záznamy vektorových dát a budúce uzly v sieti majú rozličnú reprezentatívnosť. Reprezentatívnosť je lokálna vlastnosť založená na počte objektov, ktoré sú najbližšími susedmi daného vrcholu. Lokálna reprezentatívnosť môže byť použitá na redukciu dát, v ktorej sa veľmi dobre zachová štruktúra i vlastnosti pôvodných dát. Práve preto túto vlastnosť využíva aj metóda LRNet.

Predpokladajme, že $D = (O,s)$ je dataset, kde O je množina objektov a $s : O \times O \to \mathbb{R}_{\geq 0}$ je funkcia podobnosti, ktorá kvantifikuje podobnosť medzi dvoma objektmi. Výpočet podobnosti je zvyčajne založený na Gaussovej funkcii. Objekty O_j, pre ktoré platí, že $s(O_i, O_j) > 0$ sú susedia. Najbližším susedom objektu O_i je objekt O_j s najväčšou podobnosťou. Ak je takýchto objektov viac, tak má objekt O_i viacero najbližších susedov.

Definícia 1 *(lokálny stupeň).* Lokálnym stupňom nazývame počet objektov, pre ktoré je O_i sused, budeme ho značiť ako $deg(O_i)$.

Definícia 2 *(lokálna významnosť).* Nech $g(O_i)$ je počet objektov, pre ktoré je O_i najbližší sused. Potom $g(O_i)$ je lokálna významnosť objektu O_i. Objekt nazývame lokálne významným, ak $g(O_i) > 0$, inak je lokálne nevýznamný.

Definícia 3 *(x-reprezentatívnosť).* Nech $x \in \mathbb{R}_{>1}$ a $deg(O_i) > 0$. Potom $r_x(O_i)$ je *x-reprezentatívnosť* objektu O_i definovaná nasledujúcim vzťahom:

$$r_x(O_i) = \frac{g(O_i)}{log_x(1 + deg(O_i))} \tag{8}$$

Dôležitým parametrom x-reprezentatívnosti je základ logaritmu x. Neformálne môžeme povedať, že x ovplyvňuje stupeň reprezentatívnosti objektu vo vzťahu k svojmu okoliu.

Definícia 4 *(základ x-reprezentatívnosti).* Nech $x \in \mathbb{R}_{x>1}$ a $r_x(O_i) = 1$. Potom $x = b(O_i)$ je *základ x-reprezentatívnosti* objektu O_i definovaný ako:

$$b(O_i) = (1 + deg(O_i))^{\frac{1}{g(O_i)}} \tag{9}$$

Základ x-reprezentatívnosti reprezentuje okolie x-reprezentatívnosti objektu. Menší základ x-reprezentatívnosti, znamená vyššiu lokálnu dôležitosť. Lokálne nevýznamné objekty a outlieri (outlayermi nazývame odľahlé pozorovania) nemajú žiadny základ x-reprezentatívnosti, teda nie je x, pre ktoré by lokálne nevýznamné obejkty boli x-reprezentantmi datasetu D.

Na skonštruovanie grafu môžeme využiť lokálne informácie o jednotlivých objektoch v datasete. Na základe vyššie uvedených definícií ďalej predpokladáme, že lokálna dôležitosť objektu je vo vzťahu k jeho základu x-reprezentatívnosti. Pre algoritmus konštruovania grafu si zadefinujeme ešte dva pojmy:

Definícia 5 *(lokálna reprezentatívnosť).* Nech $b(O_i)$ je *základom x-reprezentatívnosti* objektu O_i. Potom $lr(O_i)$ je *lokálna reprezentatívnosť* objektu O_i definovaná nasledovne:

$$lr(O_i) = \frac{1}{b(O_i)} \text{ pre lokálne významný } O_i, \tag{10}$$

$$lr(O_i) = 0 \text{ pre ostatné prípady,} \tag{11}$$

Definícia 6 *(reprezentatívny sused).* Nech

$$k = \text{ROUND}(lr(O_i) \cdot deg(O_i)) \tag{12}$$

a $K(O_i)$ je množina k susedov O_j objektu O_i s najväčšou podobnosťou $s(O_i, O_j)$. Potom reprezentatívnymi susedmi objektu O_i sú objekty $O_j \in K(O_i)$.

Vysvetlili sme si jednotlivé termíny, ktoré potrebujeme poznať pri metóde LRNet. Samotný algoritmus tejto metódy pozostáva z nasledujúcich krokov:

1. Vypočítanie matice podobnosti S pre dataset,
2. Vypočítanie reprezentatívnosti pre všetky objekty datasetu,

3. Priradenie jedného vrcholu každému záznamu z datasetu,

4. Vytvorenie zoznamu hrán tak, že sa vytvorí hrana medzi uzlami v_i a v_j ak O_j je najbližším susedom O_i alebo O_j je reprezentatívnym susedom O_i za predpokladu, že $i \neq j$.

2.4.5 Percento najvýznamnejších hrán

Posledná metóda implementovaná v tejto práci, je nazvaná percento najvýznamnejších hrán. Jej algoritmus každému záznamu z vektorových dát priradí uzol. Následne pre každý uzol spočíta jeho podobnosť či vzdialenosť od všetkých ostatných uzlov. Hrany zoradíme podľa toho, či bola pri vytváraní siete zvolená podobnosť alebo vzdialenosť nasledovne:

- ak bola zvolená podobnosť, hrany zoradíme podľa ich váh od najväčšej po najmenšiu,
- ak bola zvolená vzdialenosť, hrany zoradíme podľa ich váh opačným spôsobom, teda od najmenšej vzdialenosti až po najväčšiu vzdialenosť.

Vo výslednom grafe budú len tie hrany, ktoré sú najvýznamnejšie (vzdialenosť $d(X, Y)$ je najmenšia alebo podobnosť $s(X, Y)$ je najväčšia). Jediným parametrom tohto algoritmu je percento najvýznamnejších hrán p_{top}, ktorým ovplyvňujeme, koľko percent zo všetkých existujúcich hrán bude vo výslednom grafe.

2.5 Metódy vzorkovania sietí

Vzorkovanie ako také nie je nevyhnutným krokom pri prevode vektorových dát na sieť, avšak za istých okolností môže byť veľmi nápomocné pri analýze výslednej siete. Keďže niektoré z algoritmov sa s rastúcou sieťou stávajú výpočetne veľmi náročné, a teda ich prevedenie trvá neúmerne dlhú dobu. Príkladom takto výpočetne náročného algoritmu je napríklad výpočet betweenness centrality.

Vo chvíľach, keď pracujeme s veľkými sieťami, môžeme teda využiť práve vzorkovanie, ktoré umožňuje vytvoriť reprezentatívnu vzorku pôvodnej siete. To znamená, že vlastnosti siete už nemusíme počítať nad pôvodnou veľkou sieťou, ale len nad redukovanou sieťou. Existuje mnoho globálnych či lokálnych vlastností siete, ktoré nám umožňujú sieť pochopiť a tiež ju charakterizovať. O niektorých z týchto vlastností, akými sú napríklad priemer siete či globálny zhlukovací koeficient, sa môžeme dočítať aj v tejto práci v kapitole 3. Čo je dôležité zistiť, je to, či vzorka, získaná z pôvodne veľkej siete, dostatočne zachytáva vlastnosti tejto pôvodnej siete. Ak áno, prevedenie algoritmov nad vzrokou môže mať takmer rovnaký efekt s podobným výsledkom, ako by malo spustenie algoritmu nad pôvodnou sieťou, avšak s omnoho nižšou výpočetnou náročnosťou. Výhody vzorkovania je možné pozorovať aj pri vizualizácii sietí. Veľké siete sa totiž môžu stať značne neprehľadné a nevyužiteľné pre vizuálnu analýzu.

Algoritmy, ktoré umožňujú vytvoriť vzorky z pôvodnej siete nazývame vzorkovacie algoritmy. Konceptuálne môžeme vzorkovacie algoritmy rozdeliť do troch skupín [29]:

1. metódy založené na náhodnom výbere uzlov,
2. metódy založené na náhodnom výbere hrán,
3. exploračné techniky, ktoré simulujú náhodné prechádzky grafom.

V tejto práci sú bližšie špecifikované celkom tri vzorkovacie algoritmy. Dva z nich spadajú medzi metódy založené na náhodnom výbere uzlov a jeden radíme medzi metódy založené na náhodnom výbere hrán. Všetky tri zmieňované algoritmy vychádzajú z [29].

2.5.1 Random Node - RN

Prvým zo vzorkovacích algoritmov je takzvaný Random Node (známy tiež pod skratkou RN), čo môžeme preložiť ako náhodný uzol. To doslovne znamená, že do výslednej vzorky vyberáme uzly z pôvodnej siete náhodne s uniformnou pravdepodobnosťou. Parametrom vzorkovacích algoritmov vo všeobecnosti je veľkosť vzorky. Ak túto veľkosť stanovíme napríklad na 20 % z pôvodnej siete, potom každý uzol bude do vzorky vybraný s pravdepodobnosťou $p = 0{,}20$. Následne sa do vzorky zaradia všetky hrany z pôvodnej siete, ktoré spájajú uzly zaradené do vzorky. Nevýhodou tohto algoritmu, ako ukázali autori v [37], je to, že nie veľmi dobre rešpektuje mocninové rozdelenie distribúcie stupňa uzlov pôvodnej siete.

2.5.2 Random Degree Node - RDN

RDN, teda Random Degree Node, je v podstate rozšírením algoritmu Random Node. Avšak pravdepodobnosť, že uzol z pôvodnej siete bude vybraný do výslednej vzorky je proporčná k jeho stupňu. Uzly s vyšším stupňom budú do výslednej vzorky zvolené s vyššiou pravdepodobnosťou ako uzly s nižším stupňom. Rovnako ako Random Node, ani Random Degree Node celkom nerešpektuje pôvodné distribúcie stupňov z mocninového zákona. Do vzorky sa dostávajú uzly s vyšším stupňom a výsledná redukovaná sieť je hustá [29].

2.5.3 Random Edge - RE

Posledným zo vzorkovacích algoritmov uvedených v tejto práci je takzvaný Random Edge (v preklade z angličtiny náhodná hrana). Tentokrát do výslednej vzorky náhodne vyberáme hrany. Výber hrany pripojenej k vrcholu v je závislý na jeho stupni, pretože uzly s vyšším stupňom majú viac pripojených hrán, teda majú väčšiu šancu byť zaradené do vzorky. Algoritmus Random Edge má taktiež svoje nevýhody, ktorými sú najmä to, že vzorka je veľmi riedko prepojená, teda priemer takejto siete bude veľký a tiež to, že nie veľmi dobre rešpektuje komunitnú štruktúru pôvodnej siete [29].

2.6 Vizualizácia siete

Uviedli sme postupy pri práci s nekompletnými záznamami. Tiež sme si ukázali možnosti normalizácie, výpočtu vzdialenosti či podobnosti a tiež to, ako skonštruovať sieť z vektorových dát.

Jednou z motivácií pre vytváranie sietí je to, že siete samé o sebe majú istú výpovednú hodnotu a už prvotný pohľad na ne môže poskytnúť základné informácie, akými napríklad sú počet uzlov, počet hrán, počet komponent, prípadne môžeme za určitých okolností vidieť nejakú zhlukovú štruktúru.

Aby sme vizuálnou analýzou dokázali extrahovať zo siete čo najviac informácií, používame tzv. layouty. Toto anglické slovo môžeme preložiť ako rozmiestnenie uzlov a v tejto práci ho budeme používať. Layouty teda určujú, ako majú byť jednotlivé uzly umiestnené v priestore. V tejto práci budeme používať niekoľko základných layoutov. Tieto layouty sú už implementované v knižnici JUNG, ktorá je popísaná v nasledujúcich kapitolách, sú nimi:

- CircleLayout, ktorý rozmiestni uzly do tvaru kruhu, má obmedzené použitie a je vhodný skôr na siete s malým počtom uzlov,
- ISOMLayout, ktorý implementuje tzv. self-organizing map layout algorithm (čo môžeme voľne preložiť ako algoritmus na seba-organizujúce rozmiestnenie), ktorý je založený na Meyerových metódach seba-organizujúcich grafov [31],
- KKLayout, ktorý implementuje tzv. Kamada-Kawai algoritmus na rozloženie uzlov, o ktorom sa môžeme dočítať v [27],
- FRLayout, ktorý bol predstavený Thomasom Fruchtermanom a Edwardom Reingoldom v [18] a podľa ich iniciálov priezvisk dostal svoje pomenovanie.

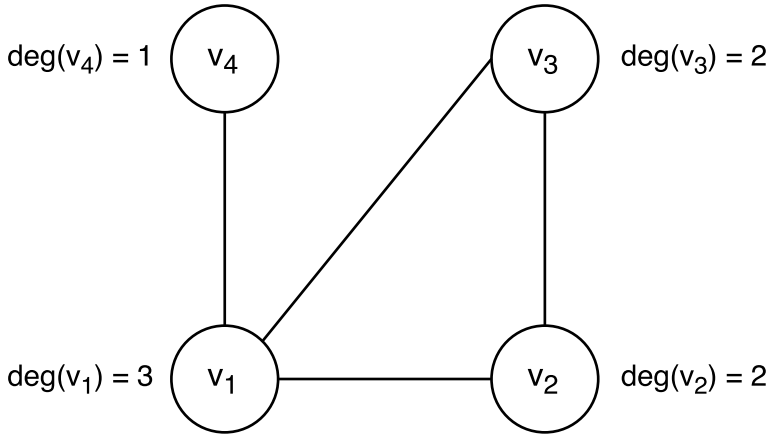
3 Vlastnosti sietí

Uviedli sme si metódy, ktoré umožňujú prevádzať vektorové dáta na sieť a tiež niektoré z možných spôsobov následnej vizualizácie novovzniknutých sietí. Hoci už samotná vizualizácia siete nám o jej vlastnostiach môže niečo prezradiť, zďaleka nestačí na to, aby sme si urobili komplexný názor o jej štruktúre. V tejto kapitole sú preto popísané základné vlastnosti, ktoré je možné zo sietí získať. Tieto vlastnosti nám umožňujú lepšie porozumieť jednotlivým sieťam a tiež poskytujú možnosť na porovnávanie rôznych sietí.

Sieť, v jej najzákladnejšej forme, je množina bodov spojených čiarami. V matematickej literatúre sieť často býva označovaná pojmom graf. Graf G je usporiadanou dvojicou $G = (V,E)$. Kde V predstavuje neprázdnu množinu vrcholov a E je množina hrán. V tejto práci sa taktiež budeme držať zavedenej konvencie označovania vrcholov a hrán, kde písmenom n budeme značiť počet vrcholov a písmenom m budeme označovať počet hrán v sieti.

3.1 Stupeň uzla

Stupeň uzla v je podľa [33] súčtom počtu k nemu pripojených hrán. Značíme $deg(v)$. Napríklad na obrázku 5, je uzol v_1 pripojený ku všetkým ostatným uzlom (v_2,v_3 aj v_4) v grafe, teda má stupeň 3. Podobne by sme mohli sledovať pripojené uzly aj pri zvyšných uzloch (tak, ako to môžeme vidieť na na obrázku 5.



Obr. 5: Ukážka výpočtu stupňa uzlov v neorientovanom grafe

Priemerný stupeň uzla v celej sieti vypočítame nasledovne:

$$deg_{avg} = \frac{1}{n}\sum_{i=1}^{n} deg(v_i) \tag{13}$$

Ľahko môžeme vypočítať priemerný stupeň z obrázka 5. Priemerný stupeň deg_{avg} = (1 + 3 + 2 + 2) / 4 = 2.

3.2 Hustota siete

Hustota siete je daná pomerom existujúcich hrán medzi uzlami voči všetkým možným potenciálnym spojeniam medzi uzlami [33].

Na výpočet hustoty je potrebné najprv zistiť počet všetkých potenciálnych spojení, ktorý si označíme P_c. Tento počet vypočítame podľa rovnice (14).

$$P_c = \frac{n * (n-1)}{2} \tag{14}$$

Keď poznáme hodnotu P_c, potom jednoducho môžeme vypočítať aj hustotu ρ podľa rovnice (15).

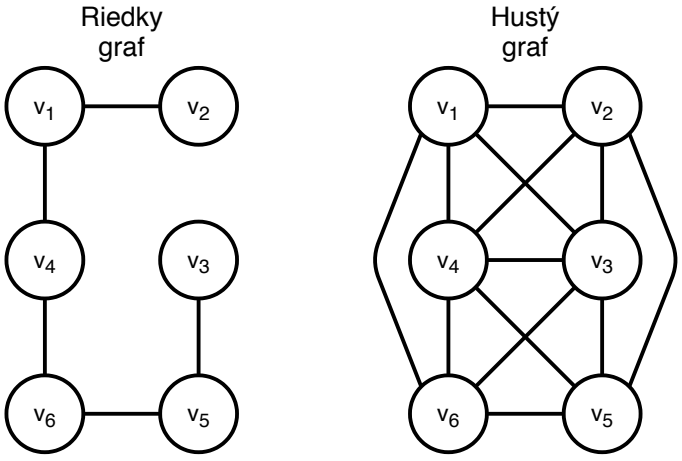
$$\rho = \frac{\mathrm{m}}{P_c} \tag{15}$$

Hodnota hustoty siete je z intervalu $0 \leq \rho \leq 1$, pretože v žiadnom grafe nie je záporný počet hrán a zároveň počet hrán, ktoré sa nachádzajú v sieti, nikdy nebude vyšší ako P_c.

Ak hovoríme o hustote siete rozlišujeme husté a riedke siete. Veľmi zjednodušene by sme mohli povedať, že riedke siete sú také, v ktorých je málo hrán a husté siete sú zas také, kde je hrán veľa. Formálnejšie môžeme uviesť, že:

- pre riedku sieť platí, že $|\mathrm{m}| = \mathrm{O}(|\mathrm{n}|)$,
- pre hustú sieť platí, že $|\mathrm{m}| = \Theta(|n|^2)$.

Obrázok 6 na jednoduchom príklade ukazuje rozdiel medzi riedkym a hustým grafom. Počet uzlov n je v oboch prípadoch 6, no v riedkom grafe je len 5 hrán, zatiaľ čo v hustom grafe ich je až 13 (kompletný neorientovaný graf na šiestich vrcholoch má celkom 15 hrán).



Obr. 6: Rozdiel medzi hustým a riedkym grafom

3.3 Lokálny a globálny zhlukovací koeficient

Názvom zhlukovací koeficient označujeme pravdepodobnosť, že dvaja susedia uzla v sú si susedmi aj navzájom. V skutočnosti nám zhlukovací koeficient meria hustotu trojuholníkov (trojuholník = cyklus na troch uzloch) vyskytujúcich sa v sieti.

V [33] je zhlukovací koeficient pre uzol v definovaný podľa nasledujúcej rovnice:

$$C_v = \frac{\text{počet párov susedov uzla v, ktoré sú spojené}}{\text{počet párov susedov uzla v}} \tag{16}$$

To znamená, že na výpočet C_v musíme prejsť cez všetky rozdielne páry uzlov, ktoré sú susedmi uzla v v danej sieti, spočítať počet takýchto párov, ktoré sú navzájom prepojené a to celé podeliť celkovým počtom párov. C_v niekedy nazývame aj lokálny zhlukovací koeficient.

Okrem lokálneho zhlukovacieho koeficientu poznáme aj globálny zhlukovací koeficient, ktorý navrhli Watts a Strogatz v [41]. Ten určeje zhlukovacieho koeficient v rámci celej siete. Vypočítame ho ako priemer lokálnych zhlukovacích koeficientov jednotlivých uzlov, tak ako to vidíme v rovnici (17).

$$C_{ws} = \frac{1}{n}\sum_{i=1}^{n} C_{v_i} \tag{17}$$

3.4 Priemer siete

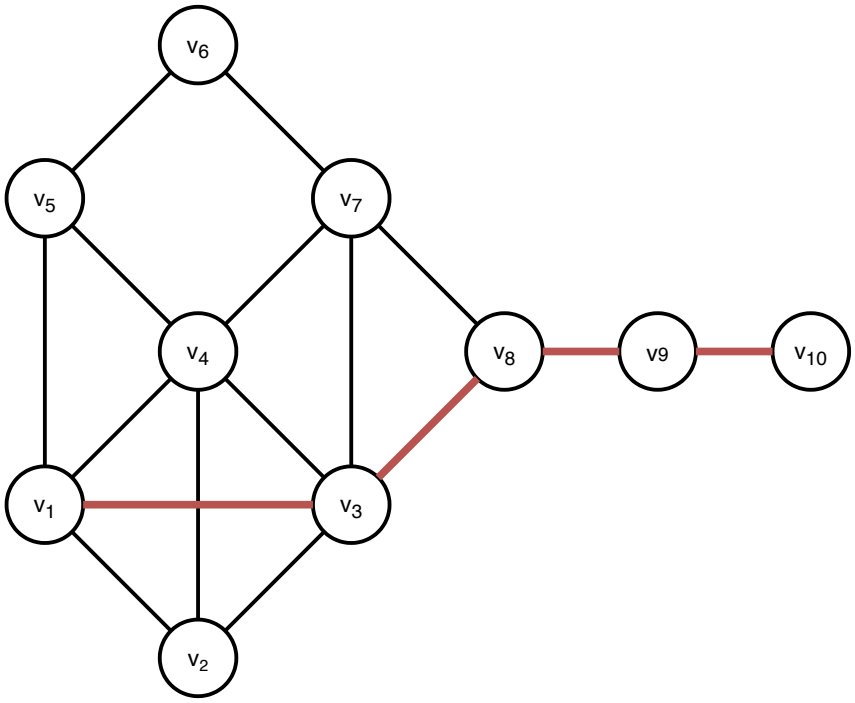
Aby sme mohli získať priemer potrebujeme rozumieť nasledujúcim pojmom z teórie grafov: sled, vzdialenosť, ťah, cesta a najkratšia cesta. Použité definície týchto pojmov pochádzajú z [3].

Sled medzi uzlami v_1 a v_2 je taká postupnosť uzlov a hrán, ktorá má začiatok v uzle v_1 a koniec v uzle v_2.

Ťahom nazývame taký sled, v ktorom sa neopakujú hrany. Zatiaľ čo cesta je sled, v rámci ktorého sa neopakujú uzly.

Vzdialenosť uzlov definujeme počtom hrán v najkratšej ceste medzi týmito uzlami. Pre vzdialenosť medzi u a v budeme používať označenie d_{uv}. Cesta s najmenšou vzdialenosťou medzi dvoma uzlami je nazývaná najkratšia cesta. Ak medzi dvoma uzlami neexistuje žiadna cesta, vzdialenosť $d_{uv} = \infty$, čo znamená, že uzol je z daného uzlu nedosiahnuteľný.

Priemer siete je najdlhšia najkratšia cesta medzi ľubovoľnými dvoma uzlami v rámci jednej komponenty siete. Priemer siete je ilustrovaný na obrázku 7, kde sme pre jednoduchosť uvažovali, že vzdialenosť medzi ľubovoľnými dvoma priamo prepojenými uzlami je rovná jednej. Na obrázku je červenou cestou vyznačený priemer siete, kde si môžeme overiť, že je to skutočne najdlhšia z najkratších ciest v sieti a je rovná dĺžke najkratšej cesty medzi $d_{v_1,v_{10}} = 4$.



Obr. 7: Priemer siete

3.5 Centrality

Pod pojmom centrality rozumieme jeden z najvýznamneších spôsobov zisťovania špecifík siete. Centrality v podstate hľadájú odpoveď na otázku: Ktoré uzly sú, z pohľadu príslušnej centrality, v danej sieti najdôležitejšie? Výsledkom je akési ohodnotenie, či ak chceme ranking jednotlivých uzlov. V tejto práci sú uvedené len niektoré centrality, podľa toho ako sú definované v [33].

3.5.1 Degree centrality

Zrejme najjednoduchšou centralitou je tzv. degree centrality. Degree (teda stupeň) uzla určuje koľko hrán je pripojených k danému uzlu. Napriek tomu o akú jednoduchú vlastnosť ide, môže byť veľmi užitočná. Napr. ak si ako uzly predstavíme ľudí a vzťahy medzi nimi budeme reprezentovať hranami, je evidentné, že uzol s najvyšším stupňom (v našom prípade človek s najviac kontaktmi) je dôležitý. Samozrejme, problémom je, že nevieme nič o tom, ako významné sú tieto kontakty samé o sebe (teda napr. či aj oni majú pripojenie na ďalších ľudí).

Degree centralitu počítame podľa rovnice (18).

$$C_v^{deg} = \frac{deg(v)}{n-1} \tag{18}$$

Pričom *deg(v)* je stupeň vrcholu, pre ktorý chceme vypočítať degree centralitu a n, ako sme si už zaviedli podľa konvencie, je počet vrcholov v sieti.

Priemernú degree centralitu pre celú sieť vypočítame nasledovne:

$$C_{avg}^{deg} = \frac{1}{n} \sum_{i=1}^{n} C_{v_i}^{deg} \tag{19}$$

3.5.2 Closeness centrality

Closeness centrality meria priemernú vzdialenosť z uzla do ostatných uzlov. Vypočítame ju rovnicou (20).

$$C_v^{clo} = \frac{1}{n-1} \sum_{v \neq u} d_{uv} \tag{20}$$

d_{uv} je vzdialenosť medzi uzlami u a v a je daná ako priemer najkratších ciest z daného uzlu do všetkých ostatných uzlov.

Z closeness centrality pre jednotlivé uzly vyjadríme priemernú closeness centralitu siete ako:

$$C_{avg}^{clo} = \frac{1}{n} \sum_{i=1}^{n} C_{v_i}^{clo} \tag{21}$$

3.5.3 Betweenness centrality

Ďalšou z centralít je betweenness centrality. Vychádza z toho, koľko najkratších ciest prechádza cez daný uzol. Samozrejme, čím viac najkratších ciest prechádza cez daný uzol, tým je tento uzol v príslušnej sieti dôležitejší. Betweenness centralitu vypočítame rovnicou (22).

$$C_v^{bet} = \sum_{u \neq v, w \neq v} \frac{p_{uw}(v)}{p_{uw}} \tag{22}$$

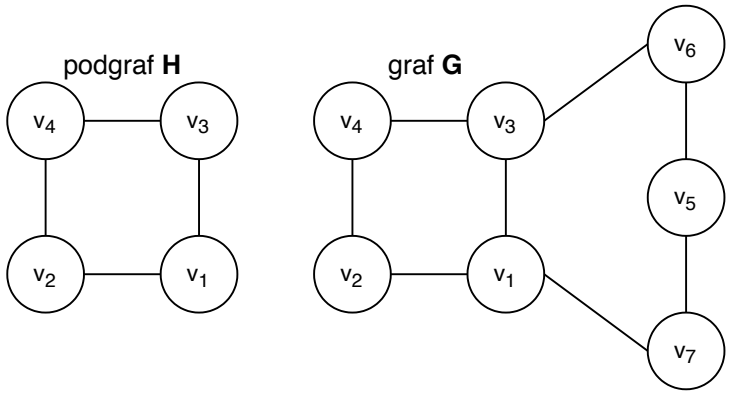
V rovnici (22) p_{uw} znamená počet najkratších ciest z uzlu u do uzla w a $p_{uw}(v)$ je počet ciest prechádzajúcich uzlom v.

Priemernú betweenness centralitu definujeme nasledovne:

$$C_{avg}^{bet} = \frac{1}{n} \sum_{i=1}^{n} C_{v_i}^{bet} \tag{23}$$

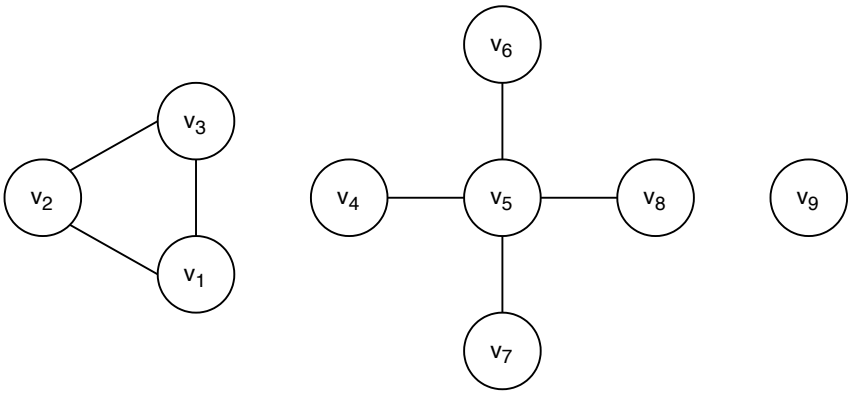
3.6 Počet komponent

Nech G je graf. Potom podľa [10] graf H je podgrafom grafu G, ak $V(H) \subseteq V(G)$ a $E(H) \subseteq E(G)$. Ukážku podgrafu môžeme vidieť na obrázku 8.



Obr. 8: Ukážka podgrafu

Súvislý podgraf H grafu G je nazývaný komponentou grafu G, ak H nie je súčasťou žiadneho súvislého podgrafu grafu G, ktorý má viac vrcholov alebo hrán ako má H [10]. Ak graf obsahuje len jednu komponentu, tak takýto graf nazývame súvislým grafom. Na obrázku 9 môžeme pozorovať graf, ktorý tvoria 3 komponenty. V rámci nesúvislých sietí môžeme počítať priemer siete, o ktorom sme sa dočítali v predošlom texte, ako najväčší priemer z pomedzi súvislých komponent, tento priemer budeme značiť priemer NC (najväčšej komponenty). Priemer siete na obrázku 9 je ∞, pretože sieť je nesúvislá. Priemer NC je potom rovný 2. V rámci prepojených komponent je najväčšia najkratšia cesta rovná práve 2 (napríklad medzi uzlami v_4 a v_7).



Obr. 9: Graf s 3 komponentami

4 Použité technológie

Najzákladnejšími úlohami výslednej aplikácie je umožniť transformáciu vektorových dát na sieť, túto sieť vizualizovať a v neposlednom rade umožniť jej analýzu. Je dôležité vytvoriť GUI (z anglického Graphical User Interface - teda grafické užívateľské rozhranie), ktoré bude pre užívateľa intuitívne ovládateľné a bude klásť dôraz na čo najväčší UX (z anglického User Experience, čo môžeme veľmi voľne preložiť ako užívateľov dojem z aplikácie). Práve vizualizácia je jedným z najdôležitejších spôsobov ako môžeme UX budovať.

Riešením, ktoré nám toto všetko umožní naplniť je kombinácia frameworkov JUNG (práca s grafmi) a JavaFX (vizualizácia). Obrovskou výhodou tejto kombinácie je, že JUNG i JavaFX sú implementované vo veľmi známom a zdokumentovanom programovacom jazyku Java, rovnako ako celé jadro finálne aplikácie. Java bola zvolená pre jej robustnosť a mnoho ďalších pozitívnych vlastností, s ktorými sa môžeme bližšie oboznámiť v [8].

4.1 JavaFX

Je zrejmé, že na to, aby bola vytvorená plnohodnotná aplikácia, je potrebné vytvorenie GUI. Ako sa môžeme dočítať v [12] rodina produktov JavaFX, ktorú vyvinuli Sun Microsystems je vhodná práve na takúto úlohu. Jadrom JavaFX je platforma Java, čo umožňuje, aby JavaFX pracovala s touto platformou bezchybne, vďaka čomu môže jednoducho ovplyvňovať existujúci Java kód.

4.2 JUNG

JUNG je skratka, ktorá vychádza z anglického Java Universal Network/Graph Framework. Ako sa môžeme dočítať v [35], JUNG je bezplatná, open-source knižnica, ktorá poskytuje bežný a rozšíriteľný jazyk na manipuláciu, vizualizáciu a tiež analýzu dát, ktoré môžu byť reprezentované ako sieť. Ako už bolo uvedné, výhodou tohto frameworku je to, že aj on je napísaný v programovacom jazyku Java, čo umožňuje aplikáciám, ktoré využívajú JUNG, aby využívali rozsiahle zabudované možnosti, ktoré poskytuje Java Application Programming Interface (API). JUNG môže tiež využívať aj už existujúce Java knižnice tretích strán a je dostupný online na adrese [39], kde taktiež môžeme nájsť aj dokumentáciu tohto frameworku.

JUNG bol navrhnutý tak, aby poskytoval podporu rozličným reprezentáciam entít a ich relácií. Za všetky uveďme neorientované a orientované grafy. JUNG ďalej zahŕňa implementácie viacerých algoritmov z analýzy sietí a vizualizácie sietí, o ktorých sa môžeme dočítať aj v tejto práci.

V ukážke kódu 1 vidíme, ako jednoducho JUNG umožňuje vytvoriť graf, pridať doň uzly a tieto uzly následne prepojiť hranou. JUNG využíva graf Graph<V,E>, kde V označuje typ vrcholu a E typ hrany. V ukážke 1 si všimneme, že v vrchol je tvorený objektom triedy Vertex a hrana objektom triedy Edge. Výhodou takejto implementácie je, že v objektoch môžeme uchová-

vať najrozličnejšie informácie, aké by jednoduché dátové typy neumožnili, čo je veľmi praktické. Vo všeobecnosti však V (vrchol) nemusí byť len objektom triedy, môže to byť aj jednoduchý dátový typ, obdobne to platí aj pre E (hranu). Všetko závisí len na tom, akú implementáciu požadujeme, aj preto je JUNG veľmi praktickým riešením.

```
// JUNG example

//initalize graph
Graph<Vertex, Edge> g = new SparseMultigraph<Vertex, Edge>();

//create vertices
Vertex v1 = new Vertex(1);
Vertex v2 = new Vertex(2);

//add vertices into graph
g.addVertex(v1);
g.addVertex(v2);

//create edge
Edge e1 = new Edge(1);

//add edge into graph and connect vertices
g.addEdge(e1, v1,v2);
```

Výpis 1: Vytvorenie grafu pridanie uzlov a hrany medzi nimi

5 Dátová sada

Táto diplomová práca je zameraná na metódy pre konštrukciu komplexných sietí z vektorových dát. To znamená, že umožňujeme skonštruovať komplexnú sieť z ľubovoľných vektorových dát. Pre analytickú časť tejto práce však bol získaný špecifický dataset, na ktorom bola prevedená aj väčšina experimentov, ktorých výsledky sú uvedené v kapitole 7. Tento dataset vychádza z medzinárodnej štúdie známej pod menom HBSC. Štúdia HBSC i dataset sú popísané v nasledujúcich podkapitolách. Rovnako je popísaný aj testovací dataset, na ktorom bola overovaná správnosť implementácie.

5.1 HBSC štúdia

Skratka HBSC vychádza z anglického The Health Behavior in School-aged Children, čo by sme mohli voľne preložiť ako zdravotné návyky školopovinných detí. V tejto práci však zostaneme pri všeobecne zaužívanej skratke HBSC.

HBSC štúdia ([13], [14], [36]) je medzinárodná výzkumná štúdia, ktorá sleduje správanie životného štýlu školopovinných detí. Projekt má počiatky na začiatku osemdesiatych rokov 20. storočia a bol následne prijatý pod záštitu WHO. Skratka WHO je akronym z anglického World Health Organization [42], čo je Svetová zdravotnícka organizácia. HBSC štúdia sa rozrástla z pôvodných troch členov (Nórsko, Fínsko, Anglicko), na desiatky krajín, medzi ktorými nechýbajú Slovenská a Česká republika. Obe krajiny sa pridali v roku 1994.

5.2 Experimentálny dataset

Ako pomerne obsiahly zdroj vektorových dát bola v práci použitá dátová sada HBSC štúdie [24] z Českej republiky z roku 2014. Dáta boli získané z dotazníkov, v ktorých odpovedali žiaci jednotlivých vekových kategóri na otázky spadajúce do nasledujúcich výskumných tém:

- nadváha a obezita,
- pohybové aktivity a sedavé správanie,
- zdravie školákov,
- socioekonomický status rodiny,
- stravovacie návyky,
- životná spokojnosť,
- školské prostredie,
- rizikové správanie (tým sa myslí napríklad fajčenie tabakových výrobkov, konzumácia alkoholu, užívanie nelegálnych drog a podobne),

- sebahodnotenie vlastnej postavy,
- šikana a násilné správanie,
- úrazy,
- dentálna hygiena,
- vzťahy (rodina a rovesníci),
- sexuálne správanie,
- využívanie sociálnych sietí,
- a iné.

Dáta sú rozdelené do troch samostatných súborov, ktoré reprezentujú jednotlivé vekové skupiny detí vo veku 11, 13 a 15 rokov. Odpovede z dotazníkov boli extrahované do podoby vektorových dát, teda na každom riadku je záznam o jednom žiakovi príslušnej vekovej kategórie. V stĺpcoch sú odpovede na jednotlivé otázky, ktoré boli uvedené vyššie. Taktiež je potrebné uviesť, že všetky dáta sú anonymizované podľa platných štandardov GDPR [40]. GDPR je všeobecne známy akronym z anglického General Data Protection Regulation, čo prekladáme ako všeobecné nariadanie o ochrane údajov. To znamená, že hoci datasety obsahujú citlivé údaje ako sú pohlavie, kraj, z ktorého je žiakova škola a podobne, tieto údaje nie sú zapísané slovne, ale je im pridelený číselný kód. Napríklad v stĺpci pohlavie nenájdeme hodnoty chlapec a dievča, ale len hodnoty 1 a 2.

Dáta sú často nekompletné, napríklad, ak sa dieťa rozhodlo na otázku neodpovedať, odpovedať zabudlo, prípadne na otázku odpovedať nevedelo, a tak zostal vo finalnom súbore príslušný stĺpec prázdny. Hodnoty v jednotlivých stĺpcoch môžeme rozdeliť na:

1. Nominálne - teda označujúce isté katogórie. Problémom nominálnych či kategoriálnych dát, že ich nevieme zoradiť, a tak ich porovnať. Avšak dáta v obdržaných súboroch sú špecifické tým, že často obsahujú vystupňované odpovede, ktoré môžeme istým spôsobom zoradiť. Napríklad ide o otázky typu *Ako často raňajkuješ vo všedné dni?* či *Koľkokrát týždenne konzumuješ ovocie?*.
2. Ordinálne - teda môžeme určiť poradie. Ordinálnych dát je v súboroch podstatne menej. Vyslovene ordinálnymi dátami sú len výška a hmotnosť detí.

5.2.1 Dataset č.1 - odpovede 11 ročných detí

Prvý dataset, ktorý obsahuje dáta extrahované z odpovedí 11 ročných detí, obsahuje 4 649 záznamov a 215 stĺpcov.

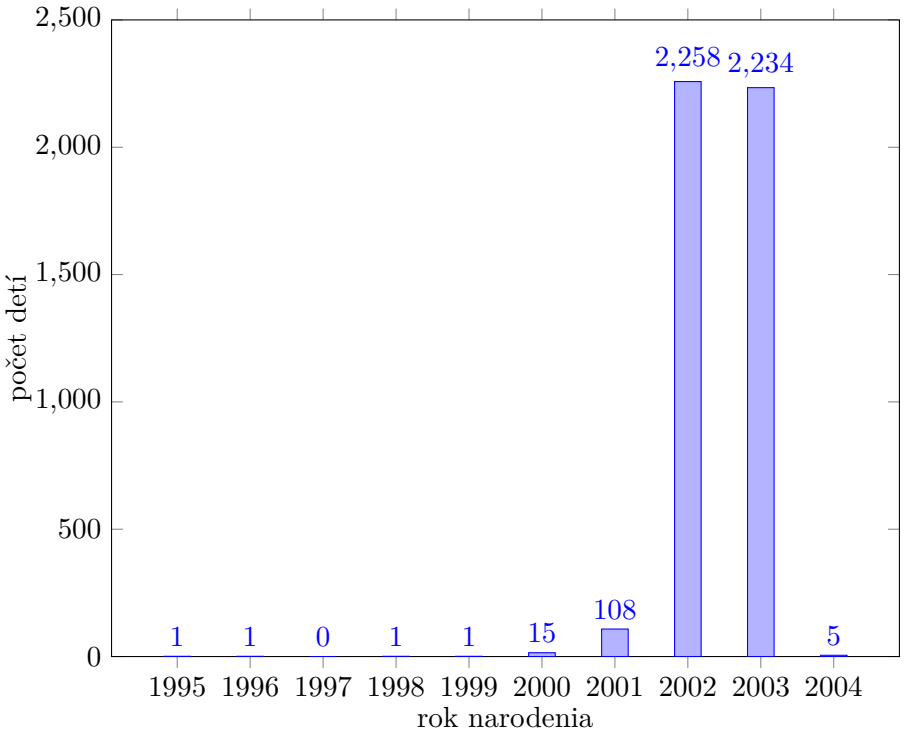
V tabuľke 1 môžeme pozorovať zastúpenie jednotlivých pohlaví. Ako bolo vysvetlené v predchádzajúcom texte, dáta sú anonymizované, a teda vieme len akési aliasy pre pohlavia, avšak vidíme, že podiel chlapcov a dievčat je približne rovnaký.

Tabuľka 1: Počet záznamov podľa pohlavia - 11 roční

Pohlavie	Počet záznamov	% (zaokrúhlené na 1 des. miesto)
1	2 298	49,4
2	2 329	50,1
neuvedené	22	0,5
Celkom	**4 649**	**100,0**

Na obrázku 10 môžeme pozorovať graf zobrazujúci počty detí podľa rokov narodenia. U 16 detí tento údaj zostal nevyplený a tak do tohto grafu neboli zahrnuté. Pripomeňme si, že dáta pochádzajú z roku 2014 a tomu zodpovedá, že drvivá väčšina detí sa narodila v rokoch 2002 a 2003. Výraznejšie je zastúpený aj rok 2001. Zaujímavosťou je, že sa v tomto súbore vyskytli aj oveľa staršie deti, napríklad máme jedno dieťa, ktoré uviedlo rok narodenia 1995. Hoci je samozrejme možné, že dieťa si v prípade neuspokojivých výsledkov ročník zopakuje, prípadne má zdravotné problémy, ktoré mu predĺžili štúdium alebo sa jednoducho pri vypĺňaní dotazníku pomýlilo, nie je potrebné tieto extrémne prípady nejako samostatne riešiť, pretože sa nachádzajú v zanedbateľných počtoch.

Obr. 10: Počet detí podľa roku narodenia - 11 roční



5.2.2 Dataset č.2 - odpovede 13 ročných detí

Druhý dataset, ktorý vychádza z odpovedí 13 ročných detí, tvorí 2 531 záznamov 296 stĺpcov.

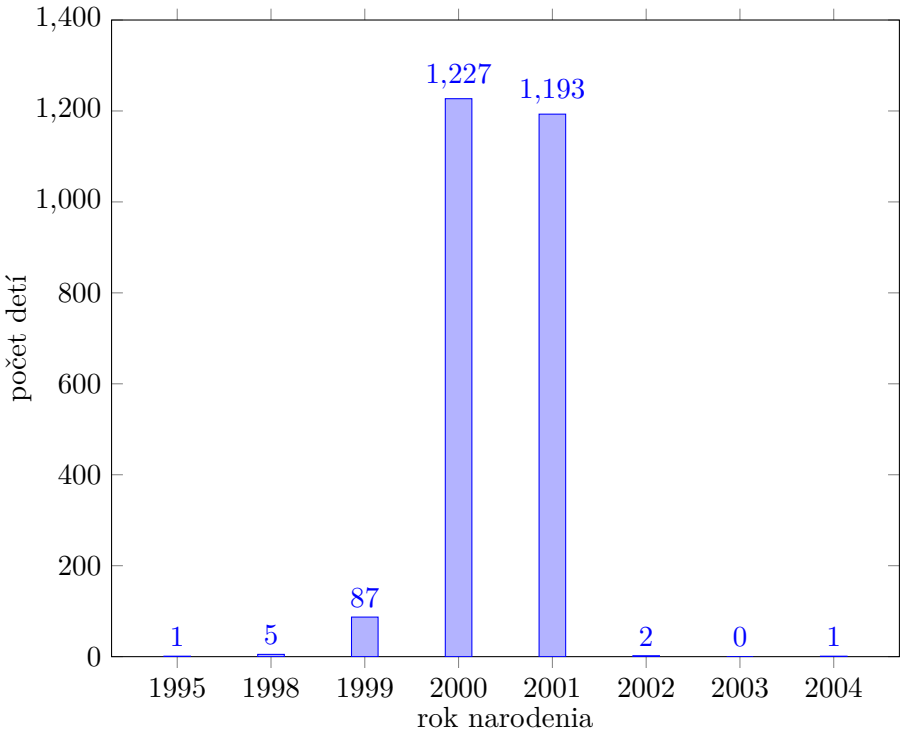
V tabuľke 2 vidíme, že oproti súboru s 11 ročnými deťmi pomerne výrazne klesol počet záznamov (detí) a naopak stúpol počet stĺpcov. To je spôsobené tým, že staršie deti majú širšiu škálu otázok a teda výsledné záznamy obsahujú aj odpovede na otázky, ktoré boli pre mladšie deti nevhodné. V tabuľke 2 tiež môžeme pozorovať zastúpenie jednotlivých pohlaví. Všimnime si, že až 15 detí malo nevyplnené pole s pohlavím.

Tabuľka 2: Počet záznamov podľa pohlavia - 13 roční

Pohlavie	Počet záznamov	% (zaokrúhlené na 1 des. miesto)
1	1 263	49,9
2	1 253	49,5
neuvedené	15	0,6
Celkom	**2 531**	**100,0**

Na obrázku 11 môžeme pozorovať graf, ktorý zobrazuje počet 13 ročných detí zotriedených podľa roku narodenia, ktoré obsahuje tento dataset. Až 15 detí neuviedlo rok narodenia, a teda nie sú zahrnuté do štatistík, ktoré sú zobrazené v grafe na obrázku 11.

Obr. 11: Počet detí podľa roku narodenia - 13 roční



5.2.3 Dataset č.3 - odpovede 15 ročných detí

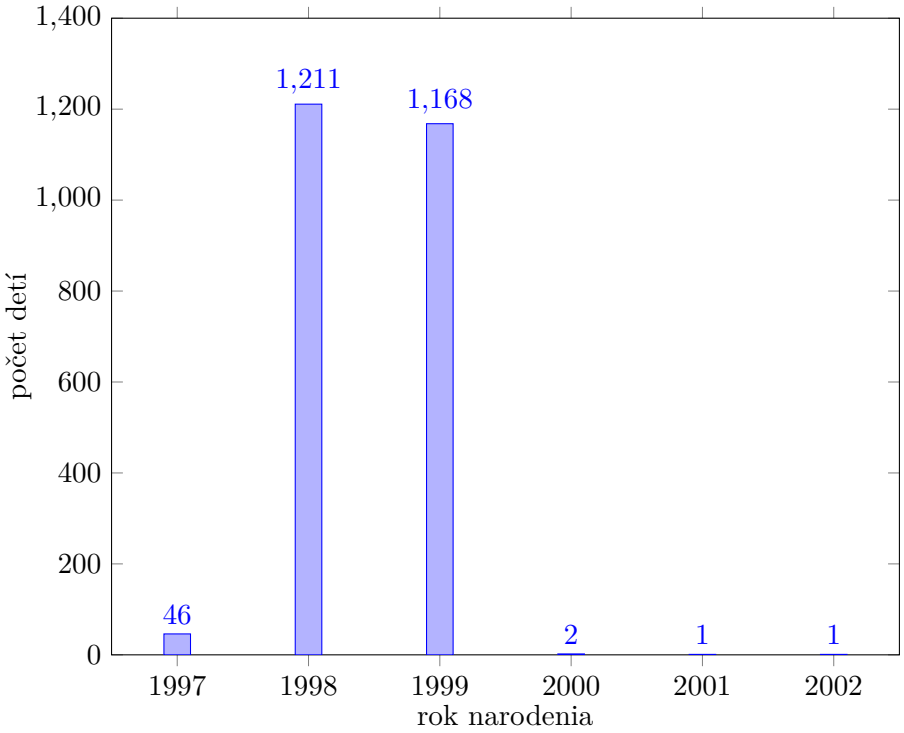
Posledný, tretí, dataset obsahuje údaje extrahované z odpovedí 15 ročných detí a tvorí ho celkom 2 432 záznamov a 319 stĺpcov. Pomer jednotlivých pohlaví je takmer vyrovnaný (pozri tabuľku 3). Počet stĺpcov je až 319, teda najvyšší zo všetkých 3 datasetov, čo je zapríčinené tým, že 15 ročné deti mohli odpovedať na väčšie množstvo otázok ako mladšie deti (najmä z oblasti užívania drog a alkoholu, sexuálnych skúseností a podobne). Počet záznamov v tomto datasete je veľmi podobný ako v datasete 13 ročných detí. Dataset s 11 ročnými deťmi má takmer toľko záznamov ako ďalšie dva dokopy.

Tabuľka 3: Počet záznamov podľa pohlavia - 15 roční

Pohlavie	Počet záznamov	% (zaokrúhlené na 1 des. miesto)
1	1 241	51,0
2	1 189	48,9
neuvedené	2	0,1
Celkom	**2 432**	**100,0**

V grafe na obrázku 12 si môžeme prezrieť počet detí zotriedených podľa roku narodenia, ktoré obsahuje tento dataset. Celkom 3 deti neuviedli rok narodenia a teda nie sú zahrnuté do štatistík v obrázku 12.

Obr. 12: Počet detí podľa roku narodenia - 15 roční



5.3 Testovací dataset

Na overovanie správnosti implementácie bol používaný dataset s názvom Iris Data Set, ktorý je dostupný na adrese [16]. Dataset je vo forme vektorových dát a tvorí ho 150 záznamov o kvetinách, ktoré sú rozdelené do 3 tried. Tieto triedy zodpovedajú špecifickému typu týchto kvetín. Dataset neobsahuje prázdne hodnoty a má 4 numerické atribúty. Analýzu týchto atribútov môžeme nájsť v [25].

6 Nástroj na konštrukciu sietí z vektorových dát

V predchádzajúcich kapitolách sme si popísali postupnosť krokov, ktorá vedie od vektorových dát k vytvoreniu siete. Ďalej sme sa zmienili o technológiách, ktoré nám umožnia tieto metódy implementovať. V nasledujúcom texte sa bližšie zoznámime s výslednou aplikáciou.

6.1 Spustenie programu a minimálne požiadavky

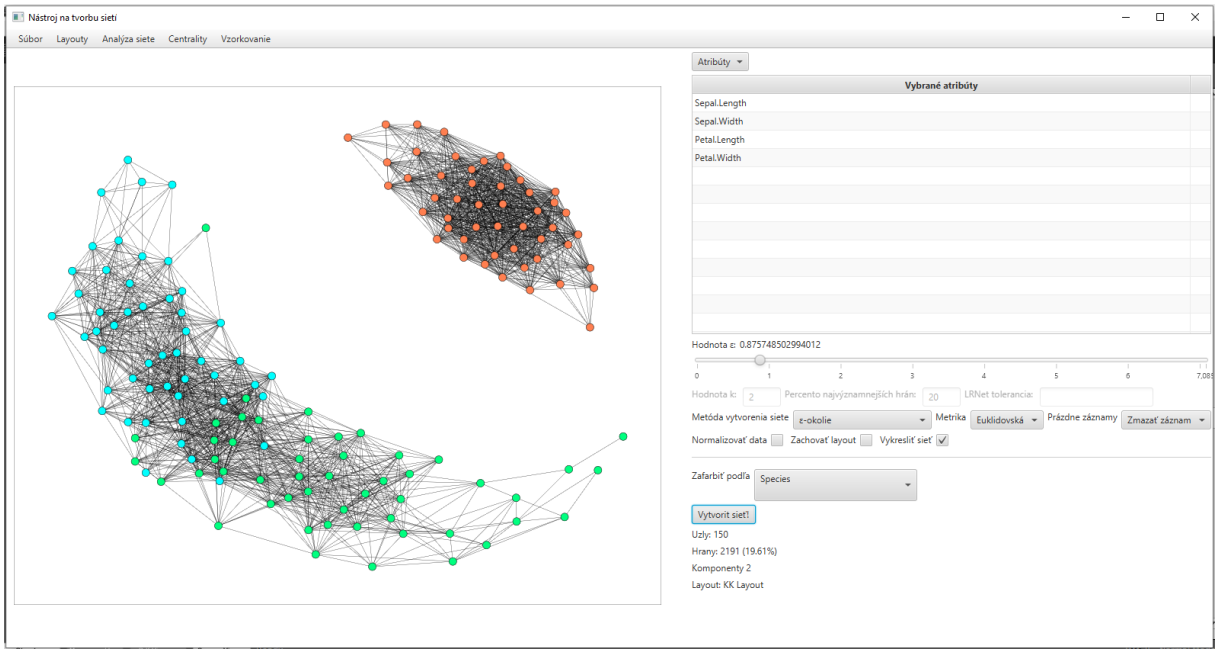
Funkčný program je uložený v prílohách tejto diplomovej práce v komprimovanej zložke s názvom *2020_URB0175_DP_priloha.zip*. Po jej rozbalení si môžeme všimnúť, že obsahuje zložku *lib*, kde sú uložené všetky externé knižnice, ktoré využíva JUNG. Program samotný spustíme tak, že dvakrát klikneme na súbor *Project2019.jar*, ktorý je hneď po spustení pripravený na používanie bez potreby inštalácie. Rozbalená zložka taktiež obsahuje zložky *src* (obsahuje zdrojový kód programu) a *datasety* (obsahuje datasety používané v tejto práci, ktoré boli popísané v kapitole 5).

Minimálnou požiadavkou je nainštalovaná Java verzia 8. Testovanie prebiehalo na notebooku s procesorom Intel Core i5 3340M. Program je optimalizovaný pre minimálne rozlíšenie HD+ (1600×900) a umožňuje aj výpočet vlastností ako sú napríklad priemer siete či closeness a betweenness centrality, ktorých výpočet so sieťou zväčšujúcou sa na radovo tisíce uzlov a státisíce hrán rastie na niekoľko desiatok sekúnd občas až niekoľko minút. Taktiež vykresľovanie sietí môže byť časovo náročné, preto program umožňuje vytvoriť sieť bez jej vykreslenia a ďalej ju analyzovať. Ako už bolo uvedené v kapitole 2.5, riešením výpočetnej náročnosti môže byť aj vzorkovanie, ktoré nám vytvorí redukované siete.

6.2 Základná obrazovka

Po spustení progamu vidíme aplikáciu, ktorej okno je rozdelené do niekoľkých častí (pozri obrázok 13). Na vytvorenie siete bol v tomto prípade použitý Iris dataset, ktorý bol popísaný v kapitole 5. Hornú lištu tvorí hlavné menu s nasledujúcimi základnými možnosťami: Súbor, Layouty, Analýza siete, Centrality a Vzorkovanie.

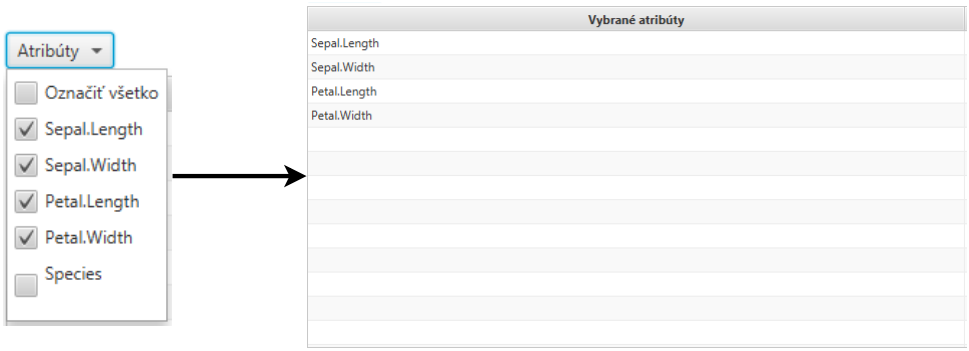
Na každú z týchto položiek hlavného menu je možné kliknúť a zobraziť tak jej ďalšie možnosti, ktoré sú popísané v tejto kapitole.



Obr. 13: Základná obrazovka programu

Ľavá časť oknovej aplikácie je vyhradená pre vizualizáciu siete. Tvorí ju tzv. canvas (plátno), v ktorom je vykreslená výsledná sieť. Pri prvotnom spustení programu je plátno prázdne.

V pravej časti vidíme viacero elementov, ktorými môžeme ovplyvňovať okolnosti za akých bude vytvorená sieť. Ako prvý je selectbox s názvom Atribúty, ktorý obsahuje checkboxy, ktoré umožňujú vybrať, ktorá podmnožina z atribútov bude použitá na vytvorenie siete. Pre prehľadnosť (najmä pri vyššom počte atribútov) sú tieto atribúty uložené v tabuľke s názvom Vybrané atribúty (pozri obrázok 14).



Obr. 14: Voľba atribútov pre vytvorenie siete

Pod tabuľkou vybraných atribútov sa nachádza slider, ktorý umožňuje nastaviť hodnotu ε. Samozrejme, to platí za predpokladu, že je v selectboxe pod ním ako Metoda vytvorenia siete zvolená metóda ε - okolie alebo kombinácia metód ε-okolie a k-NN. Inak je tento slider deaktivovaný.

Ďalej vidíme pole na vyplnenie parametru k, ktorý je aktívny pri metóde k-NN a kombinácií metód ε-okolie a k-NN. Ďalšie pole slúži na zadanie parametru pre prevod siete za využitia algoritmu Percento najvýznamnejších hrán. Posledné textové pole umožňuje nastaviť toleranciu pre algoritmus LRNet.

Okrem zvolenia metódy vytvorenia siete môžeme voliť aj metriku s akou budú počítané vzdialenosti z vektorových dát. Implementované sú metódy z kapitoly 2.3. Taktiež je možné nastaviť akciu pre nekompletné záznamy - teda pre záznamy, ktoré nemajú vyplnený niektorý z užívateľom zvolených atribútov. Aplikácia umožňuje takéto záznamy zmazať (záznamy, samozrejme, nie sú skutočne zmazané z dátového súboru iba budú vynechané pri tvorbe siete) alebo doplniť medián, prípadne doplniť priemer.

Ďalšími možnosťami ako ovplyvniť vytvorenie a vzhľad výslednej siete sú dva checkboxy. Ak zaznačíme prvý s názvom Normalizovať dáta dáta budú pred vytvorením siete normalizované za použitia normalizácie, ktorú bola popísaná v podkapitole 2.2. Ak zaškrtneme druhý checkbox s názvom Zachovať layout dané uzly zostanú rozmiestnené tak ako pri poslednom prevode. To je výhodné najmä vtedy, keď chceme vizuálne porovnať výsledky jednotlivých metód na prevod vektorových dát. Taktiež vidíme checkbox s názvom Vykresliť sieť. Vo východzom stave pri spustení aplikácie je tento checkbox zaškrtnutý, teda sieť sa bude vykresľovať. Avšak ak ho odznačíme, sieť vykreslená nebude. To je výhodné najmä pri obrovských sieťach, ktoré sú často neprehľadné a tiež to šetrí výpočtový čas potrebný na vykreslenie tisícov uzlov a často až státisícov hrán. So sieťou je naďalej možné pracovať, tak akoby bola vykreslená. Teda môžeme vypočítavať jej vlastnosti.

Poslednou možnosťou, ktorú program umožňuje je tzv. Zafarbiť podľa. V selectboxe vyberieme, podľa čoho majú byť uzly vo výslednej sieti zafarbené. Uzly môžu byť zafarbené:

1. náhodnou farbou, teda celá výsledná sieť bude obsahovať uzly jedinej farby. Je to východzie nastavenie, ktoré má v príslušnom selectboxe zvolené "Žiadne".

2. Podľa príslušnosti ku komponente. To znamená, že uzly, ktoré sa nachádzajú v spoločnej komponente, budú mať rovnakú farbu.

3. Podľa atribútu. Program umožňuje vybrať ľubovoľný atribút, podľa ktorého budú zafarbené uzly vo výslednej sieti, bez ohľadu na to v akej sú komponente. Tento prístup bol zvolený aj pri vytvorení siete na obrázku 13. Uzly sú zafarbené podľa 3 tried Iris datasetu, ktoré boli zmienené v kapitole 5.

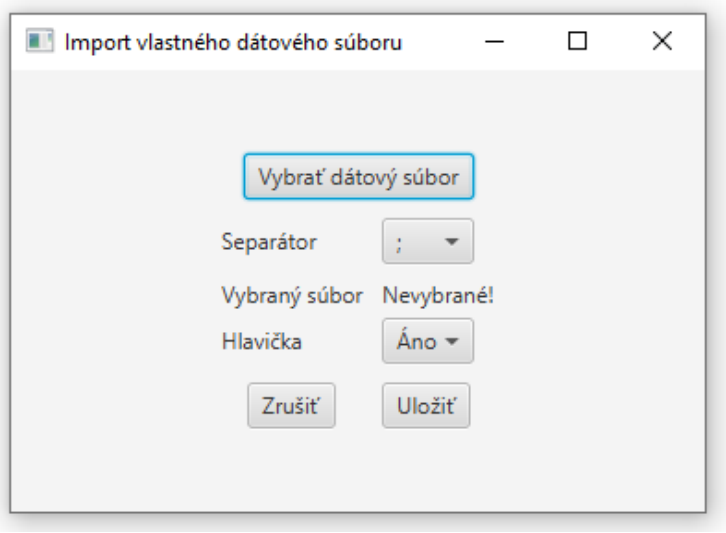
V prípade, že sme nastavili všetky elementy potrebné pre vytvorenie siete, môžeme pristúpiť k samotnému vytvoreniu siete stlačením tlačidla Vytvoriť sieť.

Po vytvorení siete sa na základnej obrazovke vykreslí sieť (ak zostal označený checkbox Vykresliť sieť) a vyplnia sa základné údaje ako počet uzlov, počet hrán, počet komponent a použitý layout, tak ako to môžeme pozorovať na obrázku 13.

6.3 Súbor

Ako už bolo uvedné, hlavné menu tvorí horná lišta programu. Prvou z jeho položiek je položka Súbor. Po kliknutí na túto položku sa zobrazia nasledujúce možnosti:

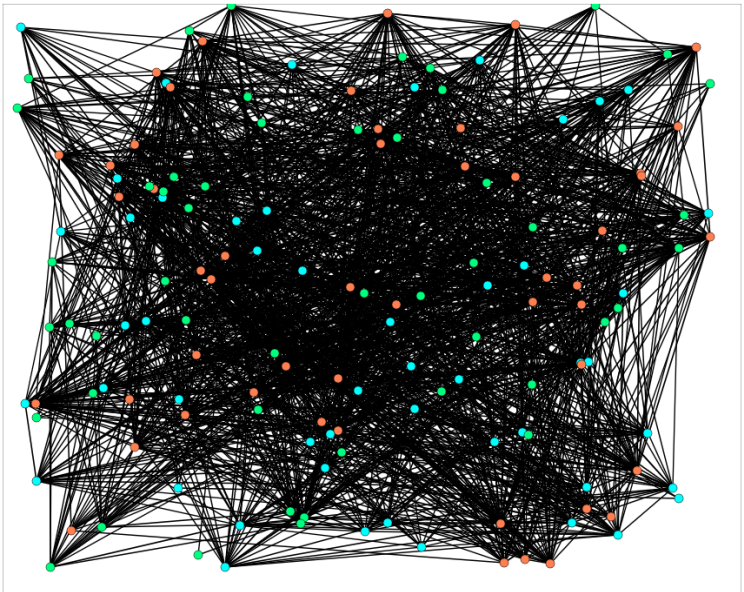
- **Exit** - ukončenie aplikácie (program je možné ukončiť aj klávesovou skratkou Ctrl+X),
- **Uložiť sieť ako GDF** - sieť bude uložená v GDF formáte, zaznamenané budú uzly a ich zafarbenie a taktiež zoznam hrán.
- **Uložiť vybrané vektorové dáta** - umožňuje uložiť dáta, ktorý boli zvolené na vytvorenie siete (na prvom riadku sú názvy zvolených atribútov a pod nimi vybrané vektorové dáta).
- **Uložiť sieť ako zoznam hrán** - uloženie siete vo forme zoznamu susedov - teda na každom riadku sú ID prepojených uzlov oddelené čiarkou.
- **Načítať dátový súbor** - po kliknutí na túto možnosť sa zobrazí nové okno (pozri obrázok 15). V tomto okne je možné nastaviť:
  1. cestu k datovému súboru s vektorovými dátami,
  2. separátor - teda symbol, ktorý oddeľuje jednotlivé atribúty vektorových dát,
  3. hlavička - umožňuje nastaviť, či súbor obsahuje hlavičku, teda pomenovanie jednotlivých atribútov. V prípade, že zvolíme možnosť Áno, zoznam atribútov sa naplní menami z hlavičky a algoritmus vytvárania siete pokračuje na druhom riadku. V prípade, že zvolíme možnosť Nie zoznam atribútov bude pozostávať z anonymných atribútov označených ako *Atribut 1, Atribut 2 ... Atribut n* v závislosti na počte atribútov, kde n je počet atribútov. Pri vytváranie siete nebude vynechaný prvý riadok ako tomu bolo pri možnosti, kde je prítomná hlavička.



Obr. 15: Importovanie vlastného súboru

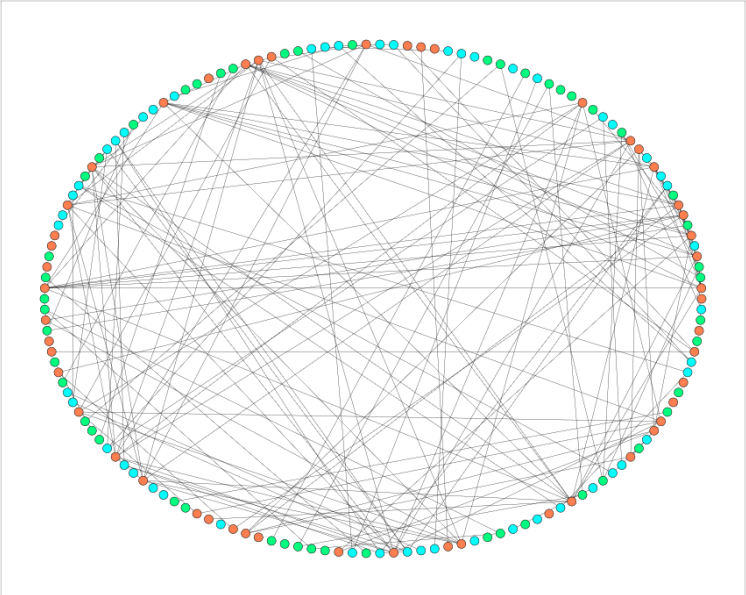
6.4 Layouty

Vysvetlili sme si, že vizualizácia je jedným z dôležitých faktorov pri budovaní UX. Keďže tento nástroj umožňuje nie len transformáciu vektorových dát na sieť, ale tiež vizualizáciu onej výslednej siete, práve tu nachádzajú uplatnenie už zmieňované layouty, ktoré rozhodujú o rozmiestnení uzlov. Prečo sú layouty dôležitou súčasťou vizualizácie si môžeme pozrieť na obrázku 16, kde nebol zvolený žiaden layout. Je zrejmé, že v takejto sieti sa orientujeme ťažko a nemá pre nás takmer žiadnu výpovednú hodnotu. Na všetky ukážky layoutov použitých v tejto kapitole bola vytvorená sieť z Iris datasetu z kapitoly 5. Vzdialenosť bola vypočítaná euklidovskou vzdialenosťou, metóda prevodu bola ε-okolie, pričom premenná ε bola nastavená na hodnotu 0,876.



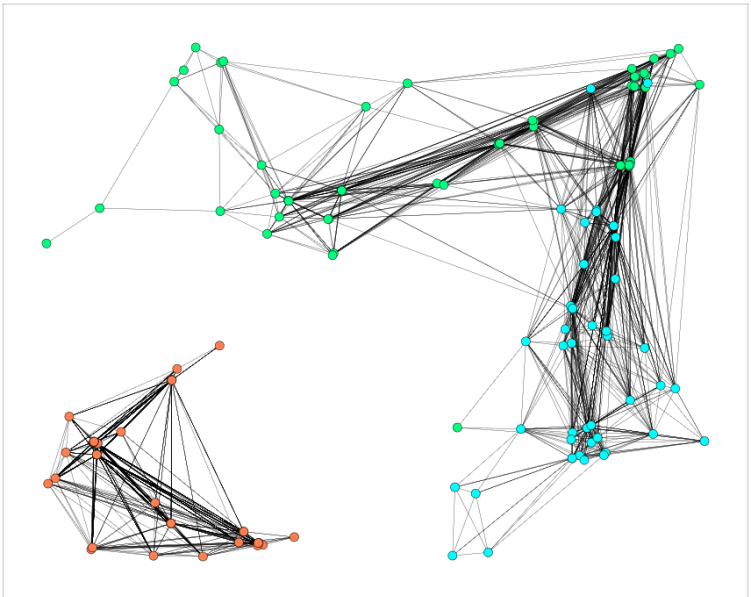
Obr. 16: Sieť bez použitia layoutu pre Iris dataset (ε=0,876)

Teraz, keď sme si ukázali význam layoutov, si v rýchlosti predstavíme tie, ktoré nájdeme v našom nástroji. Prvým je CircleLayout, ktorý rozmiestni uzly do tvaru kruhu (pozri obrázok 17). Tento layout je vhodný skôr pre siete s nižším počtom uzlov, pretože pri vyššom počte sa graf stáva pomerne výrazne neprehľadným. Hoci sieť pochádza z rovnakého datasetu ako sieť v obrázku 16, pozorujeme, že je výrazne redšia. To je spôsobené zmenšením parametru ε na hodnotu 0,286. Táto zmena bola vykonaná, aby bolo vo výslednej sieti možné rozoznať prepojenia medzi uzlami, čo bolo pri vyšších hodnotách prakticky znemožnené.



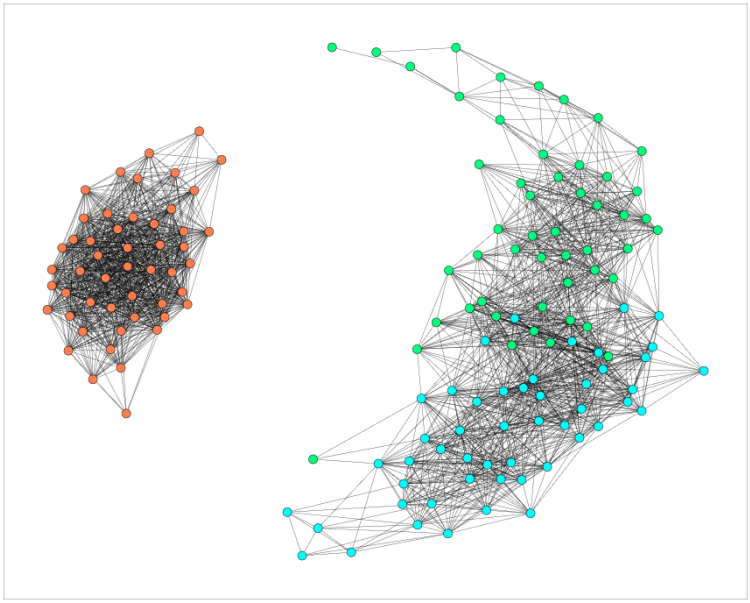
Obr. 17: Circular layout pre Iris dataset (ε=0,286)

Ďalším layoutom je ISOMLayout. Ten implementuje tzv. self-organizing map layout algorithm (teda voľne preložené algoritmus na seba-organizujúce rozmiestnenie), ktorý je založený na Meyerových metódach seba-organizujúcich grafov [31]. ISOMLayout môžeme vidieť na obrázku 18.



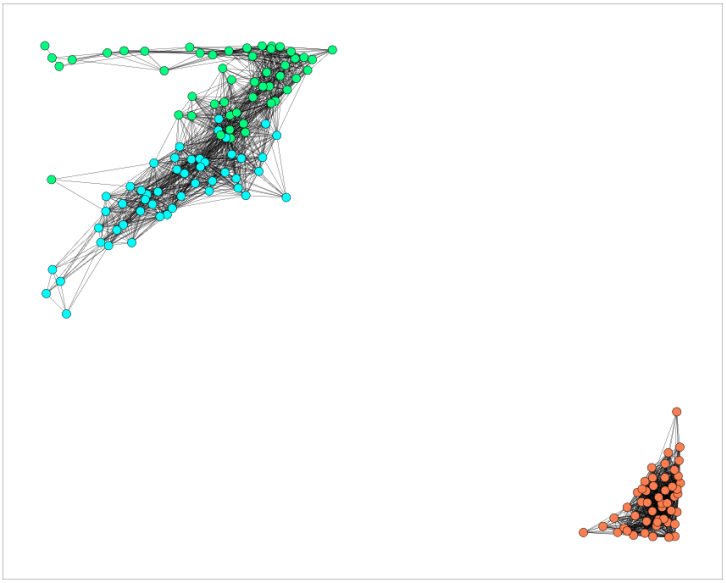
Obr. 18: ISOM Layout pre Iris dataset (ε=0,876)

Program taktiež umožňuje aj využitie KK layoutu, ktorý implementuje tzv. Kamada-Kawai [27] algoritmus na rozloženie uzlov (pozri obrázok 19).



Obr. 19: KK Layout pre Iris dataset (ε=0,876)

Posledným layoutom, ktorý je možné použiť v tejto aplikácii je tzv. Fruchterman-Reingold Layout (môžeme ho vidieť na obrázku 20).



Obr. 20: FR Layout pre Iris dataset (ε=0,876)

Napriek tomu, že layouty sú veľmi užitočným nástrojom musíme si uvedomiť, že majú tiež svoje limity. Napríklad pri sieťach, kde sa vyskytujú radovo tisíce uzlov sa sieť môže stať napriek ich využitiu neprehľadnou.

6.5 Analýza sietí

V poradí treťou záložkou je Analýza siete. Po kliknutí na túto záložku sa nám zobrazia základné možnosti na analýzu našej výslednej siete za pomoci vlastností siete (popísali sme si ich v predošlých kapitolách) akými sú:

- Priemerný stupeň uzlu,
- Priemer siete,
- Globálny zhlukovací koeficient,
- Hustota siete.

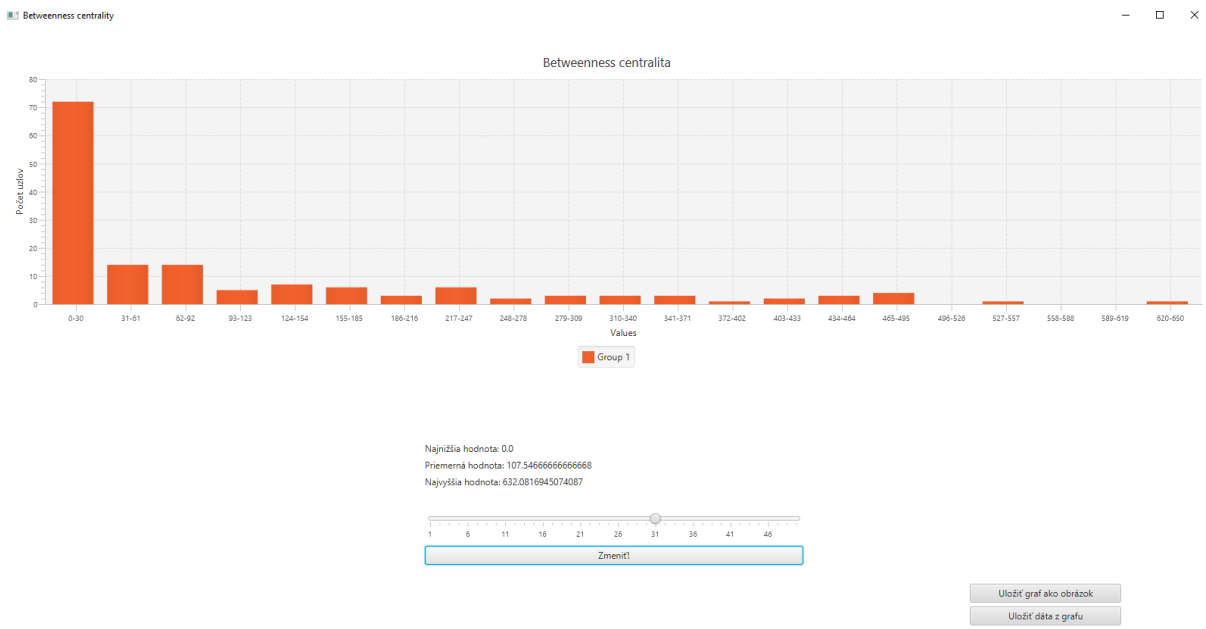
Úplne najzákladnejšie údaje o sieti, akými sú: počet uzlov, počet hrán a počet komponent sú umiestnené na základnej obrazovke aplikácie vpravo dole (pozri obrázok 13).

6.6 Centrality

Predposlednou záložkou je záložka s názvom Centrality. V tejto záložke nájdeme:

- Betweenness centralita,
- Closeness centralita,
- Degree centralita.

Po kliknutí na niektorú z týchto možností sa nám zobrazí nové okno, kde je graficky znázornená distribúcia zvolenej centrality formou histogramu. Taktiež je možné meniť intervaly histogramu ako to môžeme vidieť na obrázku 21. Rovnako na tomto obrázku môžeme pozorovať, že sa po spočítaní centralít zobrazia hodnoty minimálnej, maximálnej a priemernej hodnoty príslušnej centrality, v tomto konkrétnom prípade betweenness centrality.



Obr. 21: Betweenness centralita - zmena intervalu distribúcie

6.7 Vzorkovanie

Poslednou položkou hlavného menu je záložka vzorkovanie. Po kliknutí na ňu sa zobrazia 3 možnosti na vytvorenie redukovanej siete, ktoré boli uvedené v podkapitole 2.5. Konkrétne sú to:

- Random Node,
- Random Degree Node,
- Random Edge.

Všetky tieto algoritmy umožňujú nastaviť parameter p, ktorým ovplyvňujeme veľkosť vzorky. Zadávame hodnoty od 0 do 1, kde 1 znamená, že vzorka bude mať 100%-tnú veľkosť pôvodného grafu. Najnižšou hodnotou, ktorá bola používaná pri experimentovaní v tejto práci bola hodnota p=0,15. Možnosť nastavenia parametru p sa zobrazí po kliknutí na ľubovoľný z algoritmov.

7 Experimenty

Ukázali sme si, ako môžeme vektorové dáta previesť na sieť a ako ovplyvniť vytvorenie takejto siete nastavením najrôznejších parametrov. V tejto kapitole bude nástroj otestovaný na reálnych dátach, ktoré sme si popísali v kapitole 5. V tejto kapitole sme sa dočítali o tom, že tieto súbory obsahujú stovky rôznych atribútov, z ktorých môžeme vytvárať siete. Experimentálna časť preto bude zameraná len na vybrané podmožiny z nich.

Vo všetkých tabuľkách experimentálnej časti sú používané označenia vlasností siete definované v kapitole 3. Hustota siete v jednotlivých tabuľkách je uvedená pod označením ρ, globálny zhlukovací koeficient C_{ws}, počet komponent uvádzame skrátene pod názvom komponenty. Počet hrán je značený podľa konvencie m. Tabuľky, u ktorých to má význam tiež obsahujú záznam o počte uzlov pod označením n. Priemerný stupeň značíme deg_{avg}. Keďže mnoho sietí v experimentálnej časti bolo nesúvislých v jednotlivých tabuľkách uvádzame najväčší priemer s pomedzi jednotlivých komponent danej siete. Tento priemer označujeme Priemer NC.

7.1 Experimenty s metódami prevodu vektorových dát na siete

V tejto podkapitole je popísané, ako sa jednotlivé metódy pri rôznom nastavení ich príslušných paramterov správajú na reálnom datasete vytvorenom z odpovedí 15 ročných detí, ktorý bol bližšie popísaný v podkapitole 5.2.3. Všetky metódy budú využívať rovnakú podmnožinu atribútov. Táto podmnožina pozostáva z nasledujúcich atribútov:

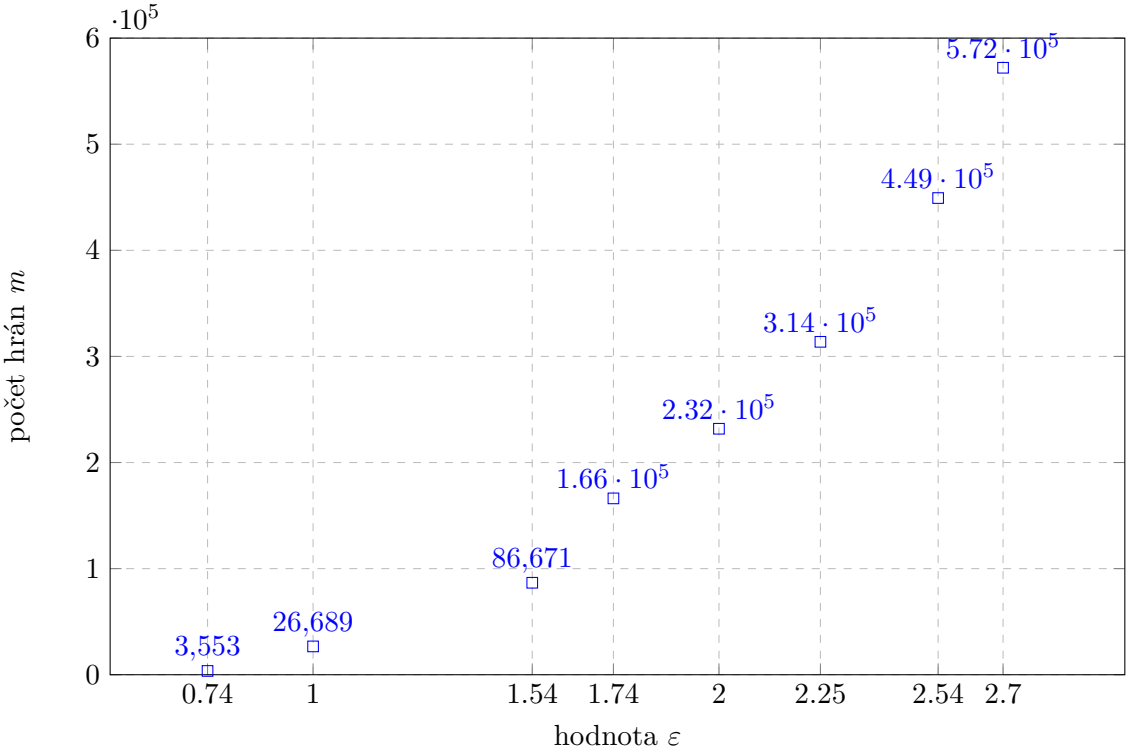
- Z129aKolikratTydneObvykleJisNeboPijesOvoce,
- Z139bKolikratTydneObvykleJisNeboPijesZeleninu,
- Z149cKolikratTydneObvykleJisNeboPijesSladkosti,
- Z159bKolikratTydneObvykleJisNeboPijesKoluNeboJineS,
- Z2314JakCastoSeStravujesVeFastFoodechmcDonaldKFC.

Názvy atribútov zodpovedajú názvom zo súborov a neboli nijako modifikované. Taktiež na výpočet vzdialenosti bola používaná rovnaká metóda a to euklidovská. V jednotlivých tabuľkách pri experimentoch s metódami prevodu vektorových dát na siete nie je uvedený údaj o počte uzlov. Je to z toho dôvodu, že vo všetkých sieťach bol počet uzlov n rovnaký (2 355). To je spôsobené tým, že všetky metódy priradzujú jednému záznamu z pôvodných vektorových dát jeden uzol vo finálnej sieti.

7.1.1 Experimenty s metódou ε-okolie

V tejto podkapitole je rozobrané, aký vplyv má hodnota parametru ε na vlastnosti výslednej siete. V grafe na obrázku 22 môžeme vidieť, ako veľmi sa zvyšuje počet hrán m v sieti (a teda aj jej hustota) so zvyšujúcou sa hodnotou ε.

Obr. 22: Závislosť počtu hrán na parametri ε



V tabuľke 4 môžeme pozorovať číselné vlastnosti sietí vytvorených pre rôzny ε. Môžeme pozorovať, že so zvyšujúcim sa ε stúpa počet hrán a teda sa zvyšuje hustota siete. Keďže pridávame stále viac hrán, nie je prekvapením ani prudký rast priemerného stupňa. Naopak počet komponent sa výrazne znižuje, pretože množstvo rozdrobených komponent pri malej hodnote ε, sa pri zvýšení tejto hodnoty spája a vytvára tak väčšie komponenty. Napriek zvyšujúcej sa hustote sietí ρ, nebola ani jedna z výsledných sietí súvislá.

Tabuľka 4: Porovnanie vlastností sietí pre rôzne nastavenie parametru ε

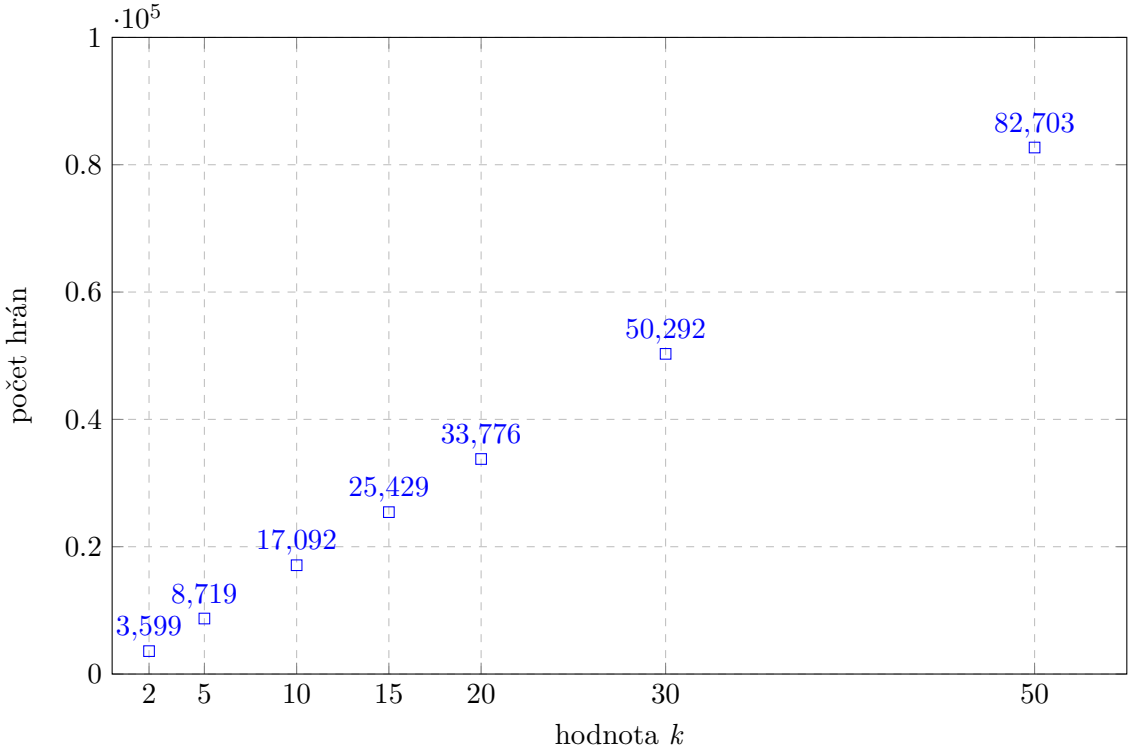
ε	**m**	**Komponenty**	$\mathbf{deg_{avg}}$	ρ	$\mathbf{C_{ws}}$	**Priemer NC**
0,741	3 553	1 297	3,02	0,0013	1,000	1
1,000	26 689	168	22,67	0,0096	0,386	25
1,540	86 671	41	73,61	0,0312	0,542	14
1,744	166 194	15	141,14	0,0600	0,562	9
2,047	231 823	10	196,88	0,0836	0,592	8
2,250	313 761	8	266,46	0,1131	0,588	8

7.1.2 Experimenty s metódou k-NN

Ako bolo uvedené v podkapitole 2.4.1, metóda k-NN má jeden parameter, ktorým je premenná k. V tejto podkapitole je názorne ukázané, ako sa výsledné siete a ich vlastnosti líšia pri rôznom

nastavení tejto premennej. V grafe na obrázku 23 môžeme sledovať, že so zvyšujúcou hodnotou k sa zvyššuje počet hrán v sieti lineárne.

Obr. 23: Závislosť počtu hrán na parametri k



V tabuľke 5 môžeme pozorovať vlastnosti pre siete vytvorené s rôznymi hodnotami premennej k. Pre hodnotu 2 mala sieť 55 komponent, pre hodnotu 5 už to boli len 2 komponenty a pre hodnotu 10 už výslednú sieť tvorila len jedna komponenta, teda graf sa stal súvislý, čo sa v experimentoch v predchádzajúcej kapitole nepodarilo napriek tomu, že počet hrán sa zvýšil na státisíce. Siete vytvorené metódou k-NN mali nižšie zhlukovacie koeficienty, ako siete vytvorené metódou ε-okolie.

Tabuľka 5: Porovnanie vlastností sietí pre rôzne k

k	**m**	**Komponenty**	**$\deg_{avg}$**	ρ	**C_{ws}**	**Priemer NC**
2	3 599	55	3,06	0,0013	0,542	33
5	8 719	2	7,40	0,0031	0,481	15
10	17 092	1	14,52	0,0062	0,443	11
15	25 429	1	21,60	0,0092	0,431	10
20	33 776	1	28,68	0,0122	0,424	9
30	50 292	1	42,72	0,0181	0,422	8
50	82 703	1	70,24	0,0298	0,439	7

7.1.3 Experimenty s kombinovanou metódou ε-okolie k-NN

V predošlých dvoch experimentálnych podkapitolách bola názorne ukázaná významnosť parametrov ε a k. Ako už bolo uvedené, kombinovaná metóda spája obe metódy do jednej výslednej metódy. V tabuľke 4 vidíme rôzne nastavenia parametru ε. Pre kombinovanú metódu boli využité len hodnoty 1,000 a 1,540 pre premennú ε z tejto tabuľky, pretože ďalšie siete s vyššou hodnotou ε už sami o sebe majú státisíce hrán.

Experimenty s kombinovanou metódou boli prevedené tak, že sa zafixovala hodnota ε a táto hodnota bola skombinovaná s rôznymi hodnotami premennej k, tak ako to môžeme pozorovať v tabuľkách 6 a 7. V oboch tabuľkách vidíme, že pridaním hodnoty k získavame takmer okamžite súvislé siete, čo sa pri samostatnej metóde ε-okolie nedarilo. Môžeme tiež pozorovať, že sa v dôsledku kombinovanej metódy mierne znížil globálny zhlukovací koeficient C_{ws}, čo môžeme pripísať tomu, že uzly, ktoré boli v pôvodnom algoritme nepripojené sa teraz pripojili ku veľkej komponente k svojim k najbližším susedom, ale inak boli stále pomerne ďaleko od zbytku uzlov v danej komponente, a teda neboli ich susedmi.

Tabuľka 6: Porovnanie vlastností sietí pre rôzne k, kde $\varepsilon = 1{,}000$

k	**m**	**Komponenty**	**$\deg_{avg}$**	ρ	**C_{ws}**	**Priemer NC**
2	27 098	3	23,00	0,0098	0,362	21
5	28 542	1	24,24	0,0103	0,362	15
10	32 317	1	27,44	0,0117	0,371	11
50	84 249	1	71,52	0,0303	0,429	7
Pôvodná sieť pre $\varepsilon = 1{,}000$						
-	26 689	168	22,67	0,0096	0,386	25

Tabuľka 7: Porovnanie vlastností sietí pre rôzne k, kde $\varepsilon = 1{,}540$

k	**m**	**Komponenty**	**$\deg_{avg}$**	ρ	**C_{ws}**	**Priemer NC**
2	86 775	3	73,68	0,0313	0,529	14
5	87 195	1	74,06	0,0315	0,518	12
10	88 480	1	75,14	0,0319	0,509	11
50	116 760	1	99,16	0,0421	0,482	7
Pôvodná sieť pre $\varepsilon = 1{,}540$						
-	86 671	41	73,61	0,0312	0,542	14

7.1.4 Experimenty s metódou LRNet

Ďalšou metódou, s ktorou boli vykonávané experimenty je metóda LRNet. V tabuľke 8 môžeme vidieť ako sa menia výsledné siete v závislosti na zmene tolerancie. Výsledné siete mali

pomerne vysoké hodnoty globálneho zhlukovacieho koeficientu C_{ws} a výsledné siete sa už pri malých zmenách tolerancie pomerne výrazne zhusťovali. Pozitívnym javom bol znižujúci sa počet komponent, ktorý predstavuje zlepšenie najmä oproti metóde ε-okolie.

Metóda LRNet na výpočet podobnosti využíva Gaussovu funkciu, definovanú v tejto práci. Preto vidíme opačný jav ako pri ostatných metódach. Teda siete sú hustejšie s nižšou hodnotou parametra. Zníženie tolerancie znamená, že za podobné považujeme aj uzly s menšou podobnosťou. Zatiaľ čo ostatné algoritmy pracujú s okolím, ktoré využíva vzdialenosť.

Tabuľka 8: Porovnanie vlastností sietí pre rôzne rôznu hodnotu tolerancie

tolerancia	**m**	**Komponenty**	**$\deg_{avg}$**	ρ	**C_{ws}**	**Priemer NC**
0,1	56 968	48	48,38	0,0206	0,487	18
0,03	86 872	20	73,78	0,0313	0,532	12
0,01	114 041	14	96,85	0,0411	0,548	13
0,005	149 273	11	126,77	0,0539	0,580	12
0,001	200 847	5	170,57	0,0725	0,604	10
0,0001	288 799	2	245,26	0,1042	0,621	10

7.1.5 Experimenty s metódou Percento najvyšších hrán

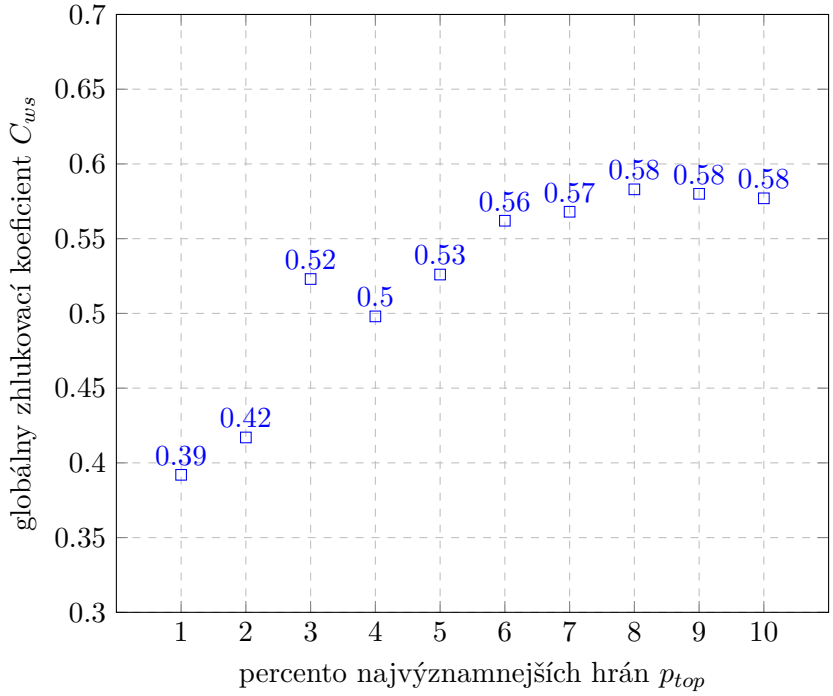
Poslednou metódou, s ktorou bolo experimentované v rámci tejto podkapitoly, vybranej dátovej sady a vybranej množiny atribútov je metóda percento najvýznamnejších hrán. Výsledky tejto metódy sú uvedené v tabuľke 9. Nastavenie parametru metódy p_{top} môžeme vidieť v prvom stĺpci, tomuto parametru zodpovedá aj hustota ρ, ktorá bola do tabuľky uvedená v rámci umožnenia rýchleho porovnania s ostatnými metódami.

Tabuľka 9: Porovnanie vlastností sietí pre rôzne percento najvyšších hrán

p_{top}	**m**	**Komponenty**	**$\deg_{avg}$**	ρ	**C_{ws}**	**Priemer NC**
1	27 719	163	23,54	0,0100	0,392	20
2	55 437	69	47,08	0,0200	0,417	15
3	83 156	41	70,62	0,0300	0,523	14
4	110 874	29	94,16	0,0400	0,498	11
5	138 592	21	117,70	0,0500	0,526	10
6	166 311	15	141,24	0,0600	0,562	9
7	194 029	10	164,78	0,0700	0,568	8
8	221 747	10	188,32	0,0800	0,583	8
9	249 466	9	211,86	0,0900	0,580	8
10	277 184	8	235,40	0,1000	0,577	8

V grafe na obrázku 24 môžeme pozorovať ako sa menil globálny zhlukovací koeficient C_{ws} so zmenou parametru p_{top}. Môžeme konštatovať, že pre testované hodnoty parametru p_{top} sa hodnota globálneho koeficientu pomerne rýchlo zvyšovala, až sa od p_{top}=6 stabilizovala na hodnote 0,58. Celkovo ale mali siete vytvorené metódou percento najvýznamnejších hrán pomerne vysoké hodnoty globálneho zhlukovacieho koeficientu. Hodnoty sú vyššie ako pri metóde k-NN a obdobné s metódou ε-okolie.

Obr. 24: Zmena globálneho zhlukovacieho koeficientu C_{ws} v závislosti na parametri p_{top}



7.1.6 Porovnanie výsledkov jednotlivých metód prevodu vektorových dát na siete

V predošlých podkapitolách sme mohli pozorovať, ako nastavenia parametrov jednotlivých metód ovplyyvňujú výsledné siete. V tejto podkapitole sú zosumarizované výsledky jednotlivých metód a je vykonané ich porovnanie. Všetky metódy sa, samozrejme, odlišujú. Preto kritériom tohto porovnania bola hustota siete ρ. Z každej metódy boli teda vybrané siete s podobnou hustotou. Výsledky pre metódu ε-okolie-okolie v tabuľke 10 boli použité z tabuľky 4, kde parameter ε bol nastavený na hodnotu 1,540. Metóda k-NN bola do tabuľky 10 vybraná z tabuľky 5, kde paramter k mal hodnotu 50. Metóda LRNet je v tabuľke 10 zastúpená výsledkami z tabuľky 8, kde parameter tolerancie bol nastavený na hodnotu 0,03.

Kombinovanú metódu ε-okolie a k-NN v tabuľke 10 vidíme pod skráteným názvom kombinovaná metóda. Do porovnania boli vybrané výsledky pre túto metódu z tabuľky 7, kde parameter ε bol nastavený na hodnotu 1,540 a parameter k mal hodnotu 2.

Metóda percento najvýznamenejších hrán je v tabuľke 10 zastúpená skratkou percento najv. hrán a boli použité namerané hodnoty z tabuľky 9. Keďže metóda percento najvyšších hrán priamo nastavuje hustotu siete, je zrejmé, že boli použité výsledky pre parameter p_{top} s hodnotou 3.

Tabuľka 10: Porovnanie výsledkov jednotlivých metód prevodu vektorových dát na siete, kde hustota $\rho \approx 0{,}0300$

Metóda prevodu	**m**	**Komponenty**	$\mathbf{deg_{avg}}$	ρ	$\mathbf{C_{ws}}$	**Priemer NC**
ε-okolie	86 671	41	73,61	0,0312	0,542	14
k-NN	82 703	1	70,24	0,0298	0,439	7
kombinovaná metóda	86 775	3	73,68	0,0313	0,529	14
LRNet	86 872	20	73,78	0,0313	0,532	12
percento najv. hrán	83 156	41	70,62	0,0300	0,523	14

V tabuľke 10 môžeme vidieť, že globálne zhlukovacie koeficienty C_{ws} jednotlivých sietí mali takmer indetické hodnoty. Výnimkou je len metóda k-NN, kde bol zhlukovací koeficient jednoznačne najnižší, čo bolo spôsobené pripojením uzlov, ktorých vzdialenosť od iných uzlov bola nedostatočná na pripojenie v iných metódach. Keďže metóda k-NN zabezpečí, aby každý z uzlov mal aspoň stupeň k, v sieti má práve tento stupeň aj veľa uzlov, ktoré sú z pohľadu ostatných vzdialené od ostatných uzlov a teda sú nepripojené. Prípadne sú pripojené len v svojej malej komponente, ktorú tvoria navzájom blízke uzly vzdialené od väčšiny ostatných uzlov. S tým súvisí aj počet komponent. Ten bol nízky len pri metóde k-NN a pri kombinovanej metóde, ktorá taktiež implementuje k-NN pre uzly so stupňom nižším ako k. Oproti metódam percento najvýznamnejších hrán a kombinovanej metóde si v tomto smere viedla významne lepšie metóda LRNet, ktorá vytvorila sieť s polovičným počtom komponent pri takmer identickej hustote siete.

7.2 Experimenty s metódami výpočtu vzdialeností a podobností

V tejto podkapitole sú zaznamenané výsledky pre jednotlivé metódy výpočtu vzdialeností a podobností, ktoré boli popísané v kapitole 2.3. Na porovnávanie jednotlivých metód bol využitý Iris dataset popísaný v kapitole 5.3. Na vytvorenie siete bola využitá metóda ε-okolie, kde ε, bol nastavený na hodnotu 0,880. Počet uzlov n, bol pre všetky siete 150 (každému záznamu z Iris datasetu bol pridelený jeden uzol vo výslednej sieti).

Metódy výpočtu podobnosti z kapitoly 2.3, ktorými sú pearsonov korelačný koefcient a gaussova funkcia, boli prevedené na nepodobnosti. Tieto metódy, tak ako boli definované v kapitole 2.3, majú hodnotu 0 do 1, kde 1 znamená najväčšiu podobnosť (objekty sú rovnaké). Aby sme previedli podobnosť na nepodobnosť, odčítame od hodnoty 1 výslednú hodnotu pearsonovho korelačného koeficientu, respektíve Gaussovej funkcie. Tým dosiahneme, že pôvodne najpodobnejšie objekty, ktoré mali výslednú hodnotu rovnú 1, sa stanú z pohľadu nepodobnosti najbližšími objektmi, kde táto nepodobnosť je rovná 0. To nám umožňuje zachovať parameter ε-okolia na

uvedenej hodnote 0,880. Napriek tomu v tabuľke 11 bola táto hodnota zachovaná len pre metódy: euklidovská, čebyševova, manhattanovská a RBF. Pearsonov korelačný koeficient pre hodnotu ε-okolia 0,880 vrátil sieť, ktorá tvorila kompletný graf, teda táto sieť prepájala navzájom všetky dvojice uzlov (celkom 11 175 hrán). Preto v tabuľke 11 bola pri použití metódy pearsonovho korelačného koeficientu zmenená hodnota parametru ε na 0,0104. Táto hodnota bola zvolená preto, aby sme dosiahli podobnú hustotu siete, ako tomu bolo pri ostatných metódach a mohli sme tak lepšie porovnať ako veľmi podobné vlastnosti majú siete vytvorené za využitia rôznych metód na výpočet vzdialenosti a podobnosti.

Tabuľka 11: Porovnanie výsledkov jednotlivých metód výpočtu vzdialeností a podobností

Metóda výpočtu	**m**	**Komponenty**	$\mathbf{deg_{avg}}$	ρ	$\mathbf{C_{ws}}$	**Priemer NC**
euklidovská	2 213	2	29,50	0,1980	0,770	8
čebyševova	3 024	2	40,32	0,2706	0,804	5
manhattanovská	831	10	11,08	0,0744	0,650	11
RBF	4 084	2	54,45	0,3655	0,877	4
pearsonov kor. koef.	3 449	2	45,98	0,3086	0,847	5

V tabuľke 11 môžeme pozorovať výsledky pre jednotlivé metódy na výpočet vzdialenosti, respektíve podobnosti. Najvýraznejší rozdiel bol zaznamenaný pri manhattanovskej vzdialenosti. Využitie tejto metódy umožnilo vytvoriť najredšiu sieť s najväčším priemerom a najnižším zhlukovacím koeficientom. Výrazne sa taktiež líšili všetky sledované hodnoty v tabuľke pre pearsnov korelačný koeficient, ktorý, ako sme si popísali vyššie, vytvoril kompletný graf a museli sme preto veľmi znížiť hodnotu parametru ε-okolie, ktorý bol pri ostatných metódach zachovaný na rovnakej hodnote. Po tejto zmene metóda pearsnovho korelačného koeficientu vytvorila podobnú sieť ako ostatné metódy. Zachoval sa vysoký priemerný stupeň, globálny zhlukovací koeficient a tiež malý priemer.

7.3 Experimenty s rôznymi súbormi

Ako bolo uvedené v kapitole 5, pre túto diplomovú prácu boli získané celkom 3 datasety. V tejto podkapitole bola preto vybraná množina atribútov, ktorá sa nachádza vo všetkých 3 datasetoch a za jej využitia sú analyzované podobnosti či odlišnosti v jednotlivých datasetoch.

Vybraná množina atribútov, ktorá bola využitá na porovnávanie jednotlivých súborov sa skladá z atribútov:

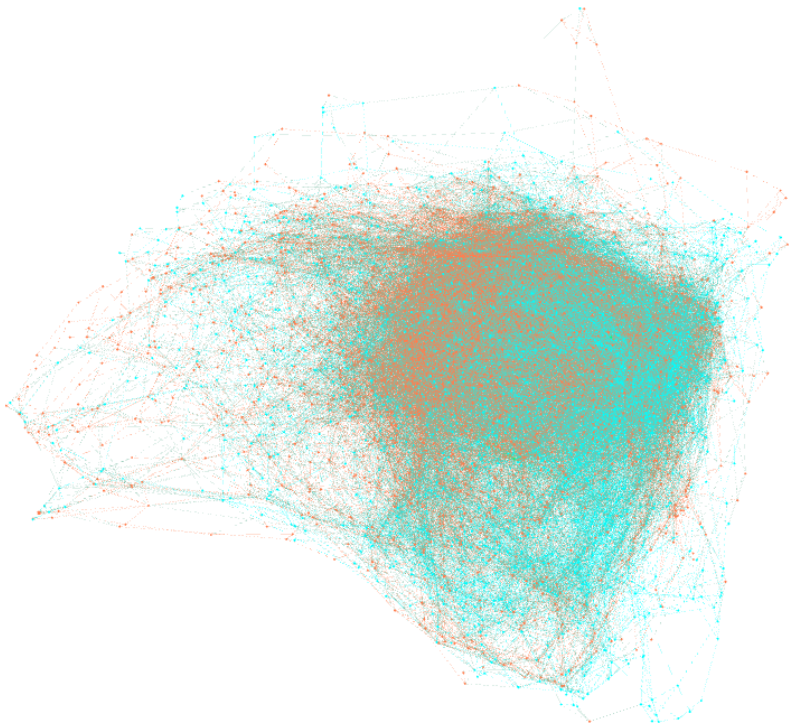
- Z96aKolikratTydneObvykleJisNeboPijesOvoce (Z129a).
- Z106bKolikratTydneObvykleJisNeboPijesZeleninu (Z139b)
- Z116cKolikratTydneObvykleJisNeboPijesSladkosti (Z149c).
- Z126dKolikratTydneObvykleJisNeboPijesKoluNeboJineS(Z159d).

- Z136eKolikratTydneObvykleJisNeboPijesMaso (Z169e).
- Z146fKolikratTydneObvykleJisNeboPijesMlekoAMlecneV (Z179f).
- Z156gKolikratTydneObvykleJisNeboPijesChipsyTycinky (Z189g).

Názvy atribútov sú ponechané bez úprav tak, ako sa vyskytujú v jednotlivých súboroch. Súbory pre 11 ročné a 13 ročné deti obsahujú zhodné názvy atribútov. Súbor pre 15 ročné deti obsahuje rovnaké otázky avšak s iným kódom, ktorý môžeme vidieť v zátvorkach pri vymenovaných atribútoch vyššie.

Na porovnanie súborov bola využitá kombinovaná metóda ε-okolie a k-NN s rôznymi nastaveniami jej parametrov. Na výpočet vzdialenosti bola použitá euklidovská vzdialenosť.

Na obrázku 25 môžeme vidieť sieť vytvorenú kombinovanou metódou s nastavením parametrov $\varepsilon = 2{,}21$ a $k = 3$. Sieť bola zafarbená podľa pohlavia dieťaťa. Kvôli anonymizácii dát, ktorá bola zmienená v úvodných kapitolách, nevieme, ktoré pohlavie má akú farbu, avšak napriek tomu, môžeme pozorovať, že jednotlivé pohlavia vytvárali navzájom pomerne husté prepojenia. Na okrajoch môžeme pozorovať uzly, ktoré boli pripojené do siete vďaka nastaveniu parametra k a bez jeho použitia to boli samostatné komponenty o veľkosti jedného uzla. Na vizualizáciu bol v tomto prípade využitý nástroj Gephi, pretože poskytoval lepšie možnosti manipulácie s vizualizovanou sieťou, ako vlastný nástroj, ktorý je popísaný v tejto práci.



Obr. 25: Vizualizácia siete 15 ročných pre $\varepsilon = 2{,}21$ a $k = 3$

V tabuľkách 12, 13, 14, 15 a 16 môžeme pozorovať ako sa líšili siete vytvorené z rovnakej množiny atribútov pre rôzne vekové kategórie detí. Názvy súborov v tabuľkách zodpovedajú vekovým kategóriám detí. Teda napr. súbor 11r obsahoval dáta o odpovediach 11 ročných detí. To, že siete vytvorené z odpovedí 11 ročných detí, obsahovali viac uzlov než zvyšné 2 súbory je spôsobené vyšším počtom záznamov v danom súbore. O počte záznamov v jednotlivých súboroch sa môžeme dočítať v kapitole 5.

Tabuľka 12: Porovnanie jednotlivých súborov pre ε=1,50 a $k = 2$

Súbor	**n**	**m**	**Komponenty**	$\mathbf{deg_{avg}}$	ρ	$\mathbf{C_{ws}}$	**Priemer NC**
11r	4 272	29 276	3	13,70	0,0032	0,343	21
13r	2 421	10 365	1	8,56	0,0035	0,326	25
15r	2 336	12 505	4	10,70	0,0046	0,329	20

V tabuľke 13 vidíme, že všetky siete boli pomerne riedke, mali nízky globálny zhlukovací koeficient a mali pomerne vysoký priemer. Všimnime si však, že zvýšenie parametru k na hodnotu 5 spôsobilo, že všetky siete boli tvorené 1 komponentou, teda boli súvislé. Narozdiel od sietí popísaných predchádzajúcou tabuľkou, kde bola z 3 sietí súvislá len jedna.

Tabuľka 13: Porovnanie jednotlivých súborov pre ε=1,50 a $k = 5$

Súbor	**n**	**m**	**Komponenty**	$\mathbf{deg_{avg}}$	ρ	$\mathbf{C_{ws}}$	**Priemer NC**
11r	4 272	33 778	1	15,82	0,0037	0,331	14
13r	2 421	13 474	1	11,14	0,0046	0,309	13
15r	2 336	15 304	1	13,10	0,0056	0,315	12

V tabuľke 14 vidíme, že zvýšenie parametru ε na hodnotu 2,00 zdvojnásobilo počet hrán v jednotlivých sieťach oproti tým zachyteným v predchádzajúcich 2 tabuľkách. Môžeme pozorovať, že opäť len 1 sieť z 3 bola súvislá.

Tabuľka 14: Porovnanie jednotlivých súborov pre ε=2,00 a $k = 2$

Súbor	**n**	**m**	**Komponenty**	$\mathbf{deg_{avg}}$	ρ	$\mathbf{C_{ws}}$	**Priemer NC**
11r	4 272	75 834	3	35,50	0,0083	0,361	16
13r	2 421	26 881	1	22,20	0,0092	0,356	20
15r	2 336	32 885	3	28,16	0,0120	0,360	16

V tabuľke 15 môžeme pozorovať, že zmenou parametra k na 5, oproti predošlej tabuľke, kde bolo k=2, sme dosiahli súvislosť sietí vytvorených zo všetkých 3 súborov. Taktiež sa nám znížil priemer týchto sietí.

Tabuľka 15: Porovnanie jednotlivých súborov pre ε=2,00 a $k = 5$

Súbor	**n**	**m**	**Komponenty**	$\mathbf{deg_{avg}}$	ρ	$\mathbf{C_{ws}}$	**Priemer NC**
11r	4 272	78 309	1	36,66	0,0086	0,357	12
13r	2 421	28 602	1	23,62	0,0098	0,348	12
15r	2 336	34 383	1	29,44	0,0126	0,355	11

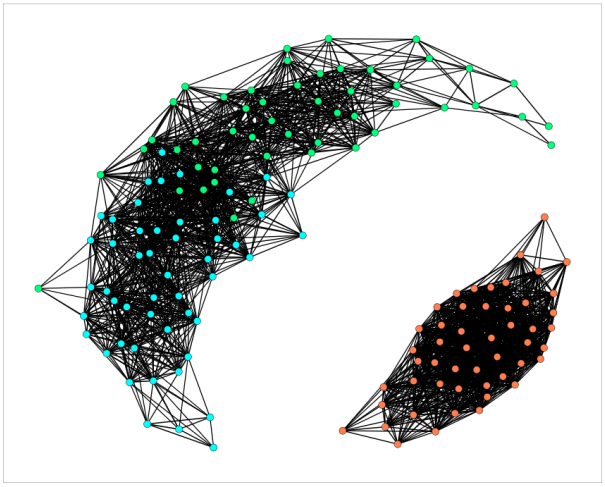
V tabuľke 16 vidíme posledné porovnanie 3 súborov s odpoveďami detí rôznych vekových kategórií. Tentokrát sme zvolili vyššie ε tak, aby sme zabezpečili vyššiu hustotu siete a k sme nastavili na 3, čím sme zabezpečili súvislosť sietí. Výsledné siete majú vyššie hodnoty globálneho zhlukovacieho koeficientu, oproti tým popísaným v predchádzajúcich tabuľkách.

Tabuľka 16: Porovnanie jednotlivých súborov pre ε=2,21 a $k = 3$

Súbor	**n**	**m**	**Komponenty**	$\mathbf{deg_{avg}}$	ρ	$\mathbf{C_{ws}}$	**Priemer NC**
11r	4 272	145 834	1	68,28	0,0160	0,431	12
13r	2 421	52 861	1	43,66	0,0180	0,432	12
15r	2 336	63 248	1	54,16	0,0232	0,442	11

7.4 Experimenty so vzorkovaním

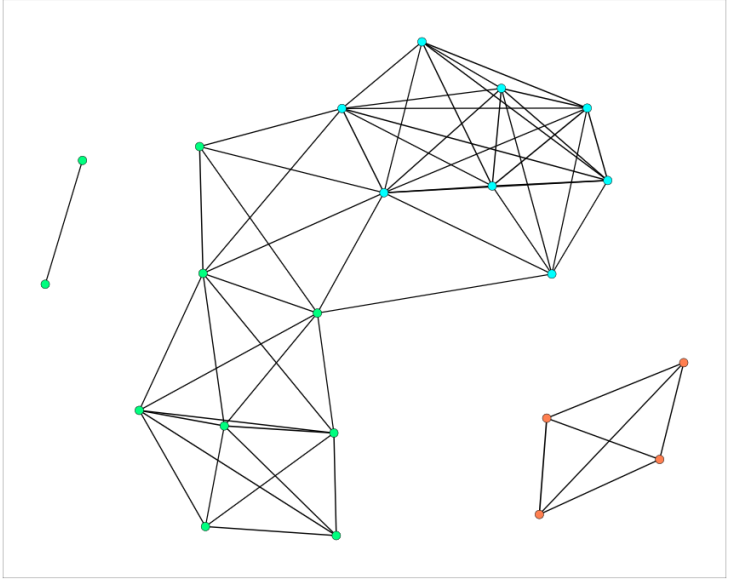
Na experimenty so vzorkovaním bol zvolený Iris dataset z kapitoly 5.3, pretože obsahuje jasne oddelené skupiny uzlov, ktoré je možné aj vizuálne jednoducho odlíšiť. Dataset bol vytvorený za využitia metódy ε-okolie, s parametrom ε nastaveným na hodnotu 0,883. Na výpočet vzdialenosti bola použitá euklidovská vzdialenosť. Pri výsledných sieťach vzniknutých zo vzorkovania bolo pri vizualizácii zachované ofarbenie uzlov podľa ich triedy, avšak layout nebol zachovaný, teda uzly nebudú na tom istom mieste ako pri pôvodnej sieti.



Obr. 26: Pôvodný Iris dataset, pre $\varepsilon = 0{,}883$

7.4.1 Experimenty s Random Node

Prvým z algoritmov, ktorý bol využitý na vzorkovanie je Random Node. Na obrázku 27 vidíme vzorku, ktorú tento algoritmus vytvoril z pôvodného datasetu s parametrom p nastaveným na hodnotu 0,15.



Obr. 27: Random Node pre p=0,15

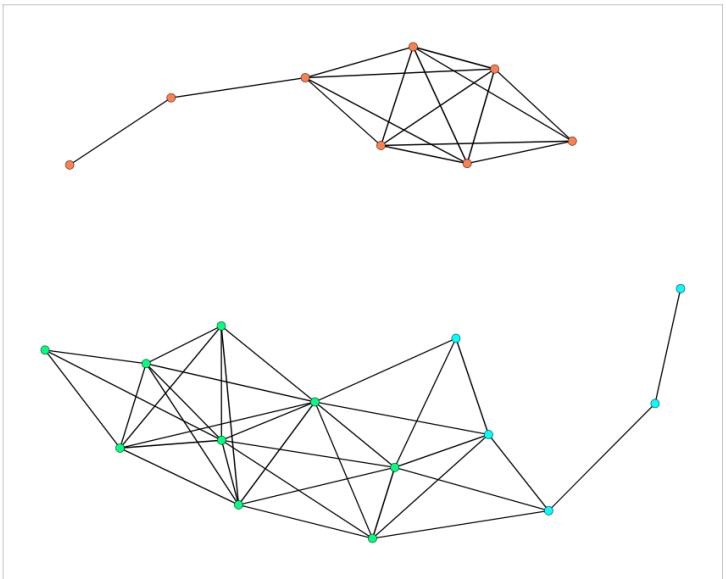
V tabuľke 17 si môžeme prezrieť ako veľmi sa líšili či zhodovali jednotlivé redukované siete oproti pôvodnému datasetu. Môžeme pozorovať, že graf sa vo viacerých prípadoch rozpadol na viac komponent, ale rozdiely v počte komponent napriek tomu neboli veľmi vysoké. Redukované siete mali s pôvodnou sieťou veľmi podobný zhlukovací koeficient a relatívne dobre sa zachovala aj hustota siete a priemer najväčšej komponenty. Najvýraznejšie sa líšili priemerné stupne, čo vyplýva z toho, že v sieti je jednoducho menej pripojení. Môžeme pozorovať, že zo zväčšujúcou sa vzorkou, sa zvyšoval aj priemerný stupeň.

Tabuľka 17: Vzorkovanie Random Node pre rôzne p

p	**n**	**m**	**Komponenty**	**$\deg_{avg}$**	ρ	**C_{ws}**	**Priemer NC**
0,15	22	58	3	5,26	0,2510	0,829	4
0,20	30	86	3	5,74	0,1977	0,740	7
0,30	45	183	2	8,12	0,1848	0,723	10
0,50	75	508	4	13,54	0,1830	0,753	9
Pôvodná sieť							
-	150	2 245	2	29,93	0,2008	0,776	7

7.4.2 Experimenty s Random Degree Node

Ďalším zo vzorkovacích algoritmov, s ktorým bolo experimentované v tejto práci je Random Degree Node. Na obrázku 28 vidíme redukovanú sieť, ktorá vznikla z pôvodného datasetu pri parametri p nastavenom na hodnotu 0,15.



Obr. 28: Random Degree Node pre p=0,15

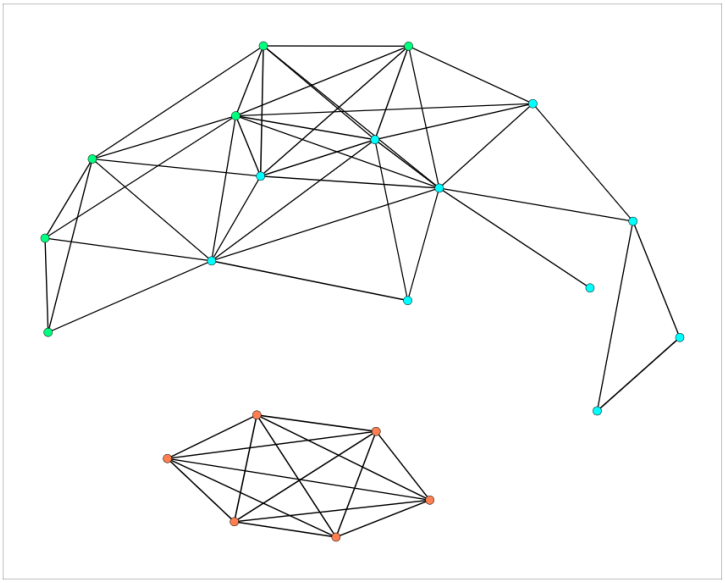
Tabuľka 18 obsahuje základné číselné vlastnosti redukovaných sietí vzniknutých za využitia algoritmu Random Degree Node. Môžeme pozorovať, že Random Degree Node vytvoril vzorky, ktoré majú vlastnosti veľmi podobné tým z pôvodného datasetu. To sa týka hlavne priemeru najväčšej komponenty, hustoty siete, počtu komponent a tiež globálneho zhlukovacieho koeficientu. Jedinou vlastnosťou, ktorá sa líšila výraznejšie je, podobne ako pri Random Node, priemerný stupeň uzla. Napriek tomu sa priemerný stupeň uzla vo vzorke z Random Degree Node s parametrom p rovným 0,50 priblížil k pôvodnej siete výraznejšie viac, ako tomu bolo pri rovnakom nastavení parametru za využitia Random Node.

Tabuľka 18: Vzorkovanie Random Degree Node pre rôzne p

p	**n**	**m**	**Komponenty**	**$\deg_{avg}$**	ρ	**C_{ws}**	**Priemer NC**
0,15	22	52	2	4,72	0,2251	0,704	5
0,20	30	111	2	7,40	0,2552	0,800	5
0,30	45	270	2	12,00	0,2727	0,830	5
0,50	75	636	2	16,96	0,2292	0,771	6
Pôvodná sieť							
-	150	2 245	2	29,93	0,2008	0,776	7

7.4.3 Experimenty s Random Edge

Posledným zo vzorkovacích algoritmov, s ktorými boli uskutočnené experimenty je Random Edge. Na obrázku 29 vidíme vzorku vytvorenú týmto algoritmom s parametrom p nastaveným podobne ako pri predošlých vzorkovacích algoritmoch na hodnotu 0,15.



Obr. 29: Random Edge pre p=0,15

Tabuľka 19 pozostáva z číselných vlastností redukovaných sietí, ktoré vznikli algoritmom Random Edge za rôznych nastavení parametru p. Počet komponent v redukovaných sieťach zodpovedal počtu komponent v pôvodnej sieti. Avšak ďalšie vlastnosti sa líšili o niečo výraznejšie. Redukované siete boli pomerne husté, z toho plynie, že ich priemery boli nižšie a tiež zhlukovací koeficient bol pri viacerých vzorkách o dosť vyšší ako ten v pôvodnej sieti.

Tabuľka 19: Vzorkovanie Random Edge pre rôzne p

p	**n**	**m**	**Komponenty**	**$\deg_{avg}$**	ρ	**C_{ws}**	**Priemer NC**
0,15	22	57	2	5,18	0,2468	0,812	4
0,20	30	126	2	8,40	0,2897	0,876	4
0,30	45	300	2	13,04	0,2898	0,796	4
0,50	75	745	2	19,86	0,2684	0,836	5
Pôvodná sieť							
-	150	2 245	2	29,93	0,2008	0,776	7

7.4.4 Experimenty s datasetom 15 ročných detí

V tejto podkapitole sú popísané výsledky jednotlivých vzorkovacích algoritmov nad datasetom 15 ročných detí. Vzorkovanie bolo vykonané nad sieťou popísanou v tabuľke 4, kde parameter

ε bol nastavený na 2,047. V tabuľke 20 môžeme pozorovať ako veľmi jednotlivé vzorkovacie altoritmy, pri rôznom nastavení parametru p, zachovali vlastnosti pôvodnej siete. V tabuľke nie sú uvedené celé názvy algoritmov, ale len ich skratky, ktoré boli zmienené v predošlých kapitolách a tiež v zozname skratiek v úvode tejto práce.

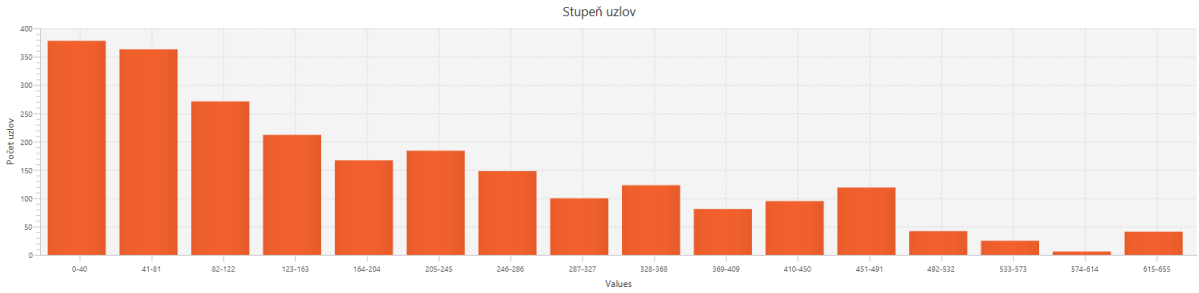
V tabuľke 20 vidíme, že algoritmus RN tvoril výrazne redšie redukované siete, ako algoritmy RDN a RE. Hustota redukovaných sietí vytvorených algoritmom RN sa výrazne približovala hustote pôvodnej siete na rozdiel od hustoty redukovaných sietí vytvorených zvyšnými dvoma algoritmami. O tom, že redukované siete vytvorené algoritmom RDN bývajú často husté, sme sa zmienili v podkapitole 2.5.2. Taktiež hodnoty globálneho zhlukovacieho koeficientu redukovaných sietí vytvorených algoritmom RN lepšie popisovali pôvodnú sieť. RE a RDN mali vyššie hodnoty globálneho zhlukovacieho koeficientu, ako pôvodná siet. Priemery siete zachytili všetky algoritmy pomerne presne. Algoritmus RN sa najvýraznejšie odchyľoval v hodnotách priemerného stupňa uzlov $\deg_{\text{avg}}$, kde naopak lepšie popisovali pôvodnú sieť algoritmy RDN a RE.

Taktiež môžeme pozorovať, že algoritmus RE tvoril redukované siete tvorené jednou komponentou, hoci v pôvodnej sieti ich bolo až 10. To zodpovedá poznatkom z kapitoly 2.5.3, kde sme si uviedli, že tento algoritmus nie veľmi dobre zachytáva komunitnú štruktúru.

Tabuľka 20: Vlastností redukovaných sietí vytvorených algoritmami RN, RE a RDN s rôznym nastavením parametru p

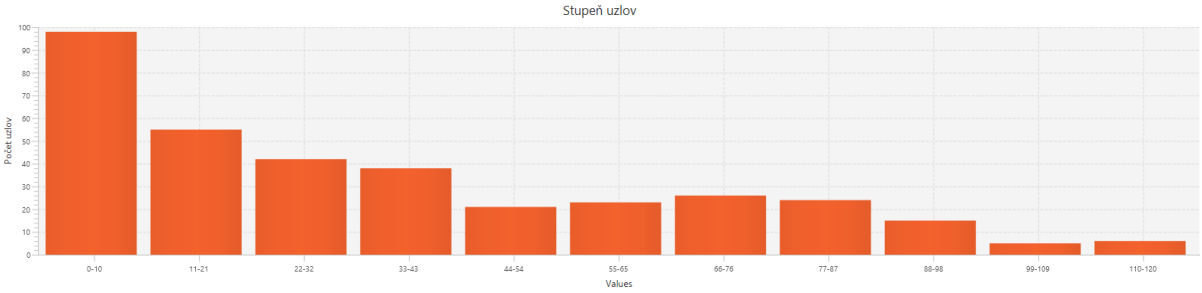
Metóda	**p**	**n**	**m**	**Komponenty**	$\mathbf{deg_{avg}}$	ρ	$\mathbf{C_{ws}}$	**Priemer NC**
RN	0,15	353	6 282	12	35,59	0,1011	0,611	9
RE	0,15	353	17 567	1	99,53	0,2828	0,703	6
RDN	0,15	353	15 719	7	89,06	0,2530	0,688	7
RN	0,20	471	9 457	11	40,16	0,0854	0,597	8
RE	0,20	471	24 574	1	104,34	0,2220	0,672	7
RDN	0,20	471	23 998	3	101,90	0,2168	0,668	7
RN	0,30	706	20 503	11	58,08	0,0823	0,589	10
RE	0,30	706	53 884	1	152,64	0,2165	0,676	8
RDN	0,30	706	54 450	3	154,25	0,2188	0,670	8
Pôvodná sieť								
-	-	2 355	231 823	10	196,88	0,0836	0,592	8

Na obrázku 30 je zobrazený histogram distribúcie stupňa uzlov pôvodnej siete vytvorenej z datasetu 15 ročných detí metódou ε-okolie s hodnotou ε=2,047. Distribúcia je v mocninovom rozdelení.



Obr. 30: Histogram distribúcie stupňa uzlov pôvodnej siete 15 ročných detí pre ε=2,047

Obrázok 31 zobrazuje histogram distribúcie stupňa uzlov redukovanej siete. Táto redukovaná sieť vznikla algoritmom Random Node (RN), kde parameter p mal hodnotu 0,15. Môžeme pozorovať, že napriek tomu, že redukovaná sieť obsahovala len 15% uzlov pôvodnej siete, distribúcia zostala veľmi podobná a taktiež zodpovedá mocninovému rozdeleniu.

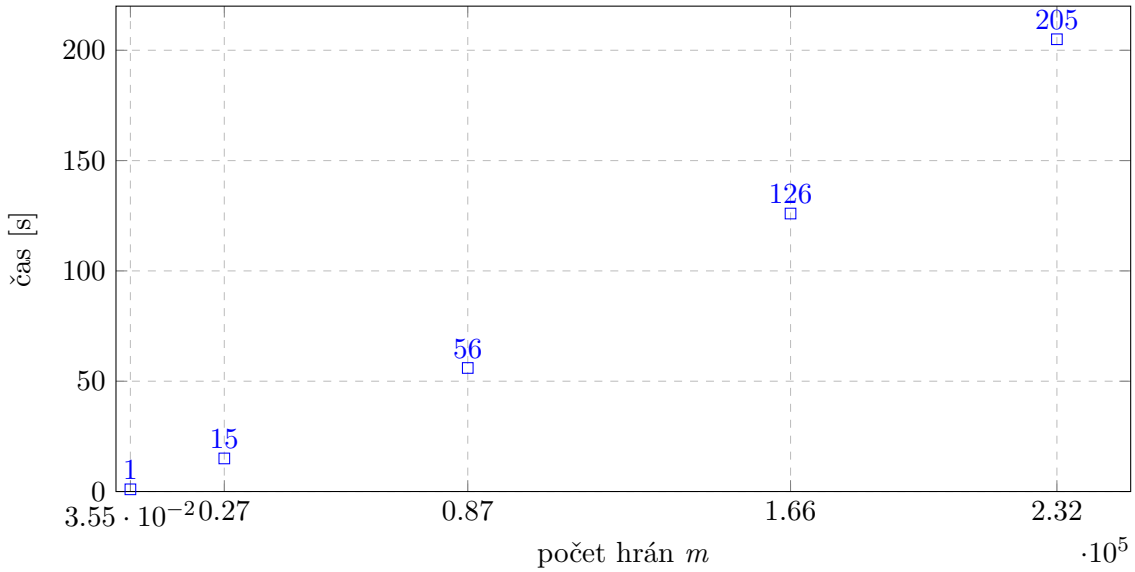


Obr. 31: Histogram distribúcie stupňa uzlov redukovanej siete 15 ročných detí vytvorenej algoritmom RN, kde p=0,15

Na obrázku 32 môžeme pozorovať ako sa menil čas potrebný na výpočet priemeru siete z tabuľky 4 so zvyšujúcim sa počtom hrán. Túto sieť tvorilo 2 355 uzlov. Vidíme, že zatiaľ čo pri cca 27 000 hranách výpočet trval 15 sekúnd. Výpočet priemeru pri sieti obsahujúcej než 232 000 hrán trval viac než 3 minúty. Práve nad touto sieťou bolo prevedené vzorkovanie.

Redukované siete, ktoré vznikli vzorkovacími metódami RN, RDN a RE, časy potrebné na výpočet priemeru siete výrazne znížili. Najhustejšia sieť, ktorá vznikla metódou RDN s parametrom p nastaveným na 0,30 vytvorila sieť so 706 uzlami a 54 450 hranami. Výpočet priemeru tejto siete trval 10 sekúnd. Výpočet priemeru pre redšie siete s nižšou hodnotou parametru p, ktoré boli zaznamenané v tabuľke 20 netrval ani 5 sekúnd. Redukované siete teda naozaj majú svoje opodstatnenie a dokážu relatívne dobre zachytiť vlastnosti pôvodnej siete a to vo výrazne kratšom čase.

Obr. 32: Časová závislosť pri výpočte priemeru siete vzhľadom na počet hrán pre grafy z tabuľky 4



8 Záver

V tejto diplomovej práci sme si uviedli celú postupnosť krokov, ktorá je potrebná pri prechode od vektorových dát k sieti. Definovali sme si metódy na výpočet vzdialenosti či podobností medzi objektmi vektorových dát, tiež možnosti normalizácie, prístup k chýbajúcim údajom a tiež metódy a ich parametre, ktoré umožňujú vytvárať siete z vektorových dát. Popísali sme si tiež niektoré zo základných pojmov z teórie grafov, ktoré nám umožňujú siete vzniknuté z vektorových dát analyzovať.

Získané vedomosti boli uplatnené pri tvorbe vlastného nástroja, ktorý implementuje zmienenú postupnosť krokov. Nástroj okrem toho umožňuje siete vizualizovať za využitia layoutov dostupných vo frameworku JUNG. Nástroj taktiež umožňuje analýzu základných vlastností sietí akými sú hustota, stupne uzlov či centrality. Ďalej nástroj umožňuje export vo viacerých formátoch, ktoré umožňujú uložiť sieť a následne ju analyzovať inými nástrojmi zmienenými v tejto práci. Nástroj bol potom využitý pri experimentálnej časti tejto práce.

V experimentálnej časti sme si overili výhody i nevýhody viacerých metód na prevod sietí a tiež to, ako čo najvhodnejšie nastaviť parametre pre tieto metódy tak, aby výsledné siete poskytovali čo najviac informácií o pôvodných vektorových dátach. Ukázalo sa, že siete tvorené metódou ε-okolie, často neboli súvislé a to napriek vysokej hustote sietí. Naopak metóda k-NN nám umožnila vytvárať súvislé siete avšak s nízkou hustotou a tiež nízkym globálnym zhlukovacím koeficientom. Tieto vedomosti sme uplatnili pri nastavení kombinovanej metódy, kde sme vhodným manipulovaním s parametrom ε zabezpečili vznik hustejších sietí s pomerne vysokými hodnotami globálneho zhlukovacieho koeficientu a následne sme navolili parameter k tak, aby tieto siete s dobrými vlastnosťami boli tvorené jednou súvislou komponentou. Taktiež sme overili, že metóda percento najvýznamnejších hrán tvorila siete veľmi sa podobajúce sieťam vytvorených metódou ε-okolie. Metóda LRNet sa ukázala ako kompromis medzi ε-okolím, respektíve percentom najvýznamnejších hrán a metódou k-NN. Keďže tvorila siete s nižším počtom komponent ako ε-okolie, ale dosahovala výrazne vyššie hodnoty zhlukovacieho koeficientu ako siete vytvorené metódou k-NN.

Experimentálne sme si tiež overili, ako sa líšia jednotlivé metódy výpočtu vzdialeností a podobností. Ukázalo sa, že napriek pomerne výrazným odlišnostiam v niektorých metódach, je možné napríklad pri využití metódy ε-okolie nastaviť jej parameter ε tak, že výsledné siete sú veľmi podobné.

Ďalej sme si popísali metódy na vytváranie redukovaných sietí, s ktorými sme následne taktiež experimentovali. Ako bolo ukázané, redukované siete nám pomáhajú znížiť výpočetnú náročnosť niektorých zo štatistických metód pri vlastnostiach sietí a tým šetria cenný čas. Tieto redukované siete taktiež umožňujú ľahšiu vizuálnu analýzu sietí, keďže stovky či tisíce uzlov sú nahradené radovo nižšími počtami vrcholov. Ukázali sme si, že redukované siete umožnujú do istej miery zachytiť vlastnosti pôvodných sietí, ako sú napríklad distribúcie stupňa uzlov. Samozrejme, redu-

kované siete majú aj svoje nevýhody. Príkladom ich nevýhod môže byť nedostatočné zachytenie vlastností pôvodnej dátovej sady, tak ako to bolo do istej miery demonštrované aj v tejto práci.

Experimentovanie s dátami vychádzajúcimi z HBSC štúdie viedlo k porovnaniu jednotlivých vekových kategórií detí. Experimentovanie bolo vedené najmä v súvislosti so stravovacími návykmi. Na vybraných podmnožinách atribútov neboli zistené žiadne významné zmeny správania, ktoré by boli dané vekom detí. Výsledné siete mali veľmi podobné charakteristiky, z čoho môžeme súdiť, že v rámci analyzovaných atribútov sa správanie deti s vekom veľmi nezmenilo. Samozrejme, najväčším vekovým rozdielom medzi deťmi v jednotlivých súboroch boli spravidla 4 roky, čo zrejme nie je dostatočne dlhá doba na to, aby sa v tomto smere zásadne zmenilo správanie veľkého počtu detí, ktoré by vyústilo vo veľké odlišnosti v rámci jednotlivých sietí.

Literatúra

[1] Mathieu Bastian, Sebastien Heymann, Mathieu Jacomy, et al. Gephi: an open source software for exploring and manipulating networks. *Icwsm*, 8:361–362, 2009.

[2] Vladimir Batagelj and Andrej Mrvar. Pajek-program for large network analysis. *Connections*, 21(2):47–57, 1998.

[3] Stefano Boccaletti, Vito Latora, Yamir Moreno, Martin Chavez, and D-U Hwang. Complex networks: Structure and dynamics. *Physics reports*, 424(4):175–308, 2006.

[4] Stephen P Borgatti, Martin G Everett, and Linton C Freeman. Ucinet. In *Encyclopedia of social network analysis and mining*, pages 2261–2267. Springer, 2014.

[5] Danah Boyd. Why youth (heart) social network sites: The role of networked publics in teenage social life. *MacArthur foundation series on digital learning–Youth, identity, and digital media volume*, pages 119–142, 2007.

[6] Danah Boyd. Social network sites as networked publics: Affordances, dynamics, and implications. In *A networked self*, pages 47–66. Routledge, 2010.

[7] Danah M Boyd and Nicole B Ellison. Social network sites: Definition, history, and scholarship. *Journal of computer-mediated communication*, 13(1):210–230, 2007.

[8] Mary Campione, Kathy Walrath, and Alison Huml. *The Java tutorial: a short course on the basics*, volume 1. Addison-Wesley Professional, 2001.

[9] Sung-Hyuk Cha. Comprehensive survey on distance/similarity measures between probability density functions. *City*, 1(2):1, 2007.

[10] Gary Chartrand. *Introductory graph theory*. Courier Corporation, 1977.

[11] Shihyen Chen, Bin Ma, and Kaizhong Zhang. On the similarity metric and the distance metric. *Theoretical Computer Science*, 410(24-25):2365–2376, 2009.

[12] Jim Clarke, Jim Connors, and Eric J Bruno. *JavaFX: developing rich Internet applications*. Pearson Education, 2009.

[13] Candace Currie, Saoirse Nic Gabhainn, Emmanuelle Godeau, International HBSC Network Coordinating Committee, et al. The health behaviour in school-aged children: Who collaborative cross-national (hbsc) study: origins, concept, history and development 1982–2008. *International Journal of Public Health*, 54(2):131–139, 2009.

[14] Candace Currie, Michal Molcho, William Boyce, Bjørn Holstein, Torbjørn Torsheim, and Matthias Richter. Researching health inequalities in adolescents: the development of the

health behaviour in school-aged children (hbsc) family affluence scale. *Social science & medicine*, 66(6):1429–1436, 2008.

[15] Michel Marie Deza and Elena Deza. Encyclopedia of distances. In *Encyclopedia of distances*, pages 1–583. Springer, 2009.

[16] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. [Online; Citované 28.04.2020 z `http://archive.ics.uci.edu/ml`].

[17] Nicole B Ellison, Charles Steinfield, and Cliff Lampe. The benefits of facebook "friends:" social capital and college students' use of online social network sites. *Journal of computer-mediated communication*, 12(4):1143–1168, 2007.

[18] Thomas MJ Fruchterman and Edward M Reingold. Graph drawing by force-directed placement. *Software: Practice and experience*, 21(11):1129–1164, 1991.

[19] Jerzy W Grzymala-Busse, Linda K Goodwin, Witold J Grzymala-Busse, and Xinqun Zheng. Handling missing attribute values in preterm birth data sets. In *International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing*, pages 342–351. Springer, 2005.

[20] Jerzy W Grzymala-Busse and Ming Hu. A comparison of several approaches to missing attribute values in data mining. In *International Conference on Rough Sets and Current Trends in Computing*, pages 378–385. Springer, 2000.

[21] Mark S Handcock, David R Hunter, Carter T Butts, Steven M Goodreau, and Martina Morris. statnet: Software tools for the representation, visualization, analysis and simulation of network data. *Journal of statistical software*, 24(1):1548, 2008.

[22] Robert A Hanneman and Mark Riddle. Introduction to social network methods, 2005.

[23] Derek Hansen, Ben Shneiderman, and Marc A Smith. *Analyzing social media networks with NodeXL: Insights from a connected world.* Morgan Kaufmann, 2010.

[24] HBSC. Hbsc | health behaviour in school-aged children, 2019. [Online; Citované 22.04.2020 z `https://hbsc.cz/ostudii`].

[25] Patrick S Hoey. Statistical analysis of the iris flower dataset. *University of Massachusetts At Lowell, Massachusetts*, 2004.

[26] Anil Jain, Karthik Nandakumar, and Arun Ross. Score normalization in multimodal biometric systems. *Pattern recognition*, 38(12):2270–2285, 2005.

[27] Tomihisa Kamada, Satoru Kawai, et al. An algorithm for drawing general undirected graphs. *Information processing letters*, 31(1):7–15, 1989.

[28] Vito Latora and Massimo Marchiori. Vulnerability and protection of infrastructure networks. *Physical Review E*, 71(1):015103, 2005.

[29] Jure Leskovec and Christos Faloutsos. Sampling from large graphs. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636, 2006.

[30] Alexandra Marin and Barry Wellman. Social network analysis: An introduction. *The SAGE handbook of social network analysis*, 11, 2011.

[31] Bernd Meyer. Self-organizing graphs—a neural network perspective of graph layout. In *International Symposium on Graph Drawing*, pages 246–262. Springer, 1998.

[32] Daniel Miller. Social networking sites. *Digital anthropology*, pages 146–61, 2012.

[33] Mark Newman. *Networks: an introduction*. Oxford university press, 2010.

[34] Eliska Ochodkova, Sarka Zehnalova, and Milos Kudelka. Graph construction based on local representativeness. In *International Computing and Combinatorics Conference*, pages 654–665. Springer, 2017.

[35] Joshua O'Madadhain, Danyel Fisher, Padhraic Smyth, Scott White, and Yan-Biao Boey. Analysis and visualization of network data using jung. *Journal of Statistical Software*, 10(2):1–35, 2005.

[36] Chris Roberts, John Freeman, Oddrun Samdal, Christina Warrer Schnohr, ME De Looze, S Nic Gabhainn, Ron Iannotti, Mette Rasmussen, International HBSC Study Group, et al. The health behaviour in school-aged children (hbsc) study: methodological developments and current tensions. *International journal of public health*, 54(2):140–150, 2009.

[37] Michael PH Stumpf, Carsten Wiuf, and Robert M May. Subnets of scale-free networks are not scale-free: sampling properties of networks. *Proceedings of the National Academy of Sciences*, 102(12):4221–4224, 2005.

[38] Partha Pratim Talukdar. Topics in graph construction for semi-supervised learning. 2009.

[39] The JUNG Framework Development Team. Jung - java universal network/graph framework, 2020. [Online; Citované 25.04.2020 z `http://jung.sourceforge.net`].

[40] Paul Voigt and Axel Von dem Bussche. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 2017.

[41] Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393(6684):440–442, 1998.

[42] WHO. Who | world health organization, 2020. [Online; Citované 22.04.2020 z `https://www.who.int`].